DISEÑO E IMPLEMENTACIÓN DE UN SISTEMA DE VISIÓN ARTIFICIAL PARA DETECTAR UNA ESPECIE DE ARÁCNIDOS Y NUEVOS BROTES DE FLORA EN UN HÁBITAT ALTERADO POR UN INCENDIO.

MAURICIO ENRIQUE MARTÍNEZ LIBREROS
JONATHAN POTES BLANDÓN

UNIVERSIDAD AUTÓNOMA DE OCCIDENTE
FACULTAD DE INGENIERÍA
DEPARTAMENTO DE AUTOMÁTICA Y ELECTRÓNICA
PROGRAMA INGENIERÍA ELECTRÓNICA
SANTIAGO DE CALI
2013

DISEÑO E IMPLEMENTACIÓN DE UN SISTEMA DE VISIÓN ARTIFICIAL PARA DETECTAR UNA ESPECIE DE ARÁCNIDOS Y NUEVOS BROTES DE FLORA EN UN HÁBITAT ALTERADO POR UN INCENDIO.

MAURICIO ENRIQUE MARTÍNEZ LIBREROS

JONATHAN POTES BLANDÓN

Proyecto de Grado ara optar el título de
Ingeniero Electrónico y Telecomunicaciones

Director
JAIME ANDRÉS ARTEAGA
Magister en Ingeniería Electrónica

UNIVERSIDAD AUTÓNOMA DE OCCIDENTE
DEPARTAMENTO DE AUTOMÁTICA Y ELECTRÓNICA
FACULTAD DE INGENIERÍA
PROGRAMA INGENIERÍA ELECTRÓNICA
SANTIAGO DE CALI
2013

Nota de aceptación:

Aprobado por el Comité de Grado en cumplimiento de los requisitos exigidos por la Universidad Autónoma de Occidente para optar al título de Ingeniero Electrónico y telecomunicaciones.

JUAN DIEGO PULGARIN GRIALDO
Jurado

JUAN CAMILO ACOSTA MEJIA
Jurado

Santiago de Cali, 22 de Octubre de 2013

AGRADECIMIENTOS

Agradecemos primeramente a Dios, por habernos dado la fortaleza, la sabiduría y los recursos para poder terminar esta etapa de nuestras vidas, agradecemos a nuestras familias, pilares de orgullo y ejemplo, por habernos apoyado durante todo este proceso, les damos las gracias a nuestro director de tesis Jaime Andrés Arteaga por su tiempo y ayuda así mismo al Ingeniero Diego Martínez por su dedicación y apoyo.

Agradecemos a todos nuestros profesores que nos brindaron sus conocimientos en las diferentes materias que vimos a lo largo de la carrera, a nuestros compañeros, gracias por su apoyo y paciencia.

Por ultimo gracias a todas las personas con las que tuvimos el privilegio de compartir y que nos brindaron su apoyo incondicional.

CONTENIDO

LISTA DE TABLAS

LISTA DE FIGURAS

LISTA DE ANEXOS

RESUMEN

Todos los entornos naturales en los que existe la presencia de grandes masas de vegetación, corren el riesgo de sufrir un incendio forestal, el cual afecta severamente no solo el equilibrio local del ecosistema sino también al medio ambiente en general. Para entender el impacto de este tipo de desastres es necesario conocer el periodo de tiempo que tarda la zona afectada en empezar a recuperarse. Una de las formas que existe para determinar la recuperación de un hábitat es comprobar la presencia de arañas que vagan en el suelo o brotes de vegetación. Por esta razón es de vital importancia para los que estudian el tema tener un registro de apariciones de esas formas de vida.

Sin embargo, realizar un estudio de estas características resulta difícil y tedioso porque es necesario monitorear constantemente el área afectada, la cual por lo general es una zona apartada y de difícil acceso.

Este proyecto busca automatizar la detección de arañas del tipo Lycosidae y de brotes de flora, por medio del diseño de un sistema embebido basado en tecnología de visión artificial que registra y envía, vía red, en caso de haber detección positiva una imagen del individuo identificado ya sea araña o brote de vegetación.

El sistema utiliza un esquema de visión artificial que consiste en la adquisición de imágenes, preprocesamiento de la imagen obtenida para mejorar su calidad, segmentación de objetos presentes en la escena, extracción de características de los objetos hallados, identificación por coincidencia de características y comunicación de resultados positivos. Todo este sistema se implementa en una tarjeta Beagle Board-XM, utilizando algoritmos de procesamiento de imágenes basados en la librería optimizada para el procesamiento de imágenes OpenCV.

Como resultado se tendrá un prototipo capaz de identificar y publicar en la web las imágenes detectadas de la araña, se reportarán los resultados de los algoritmos de procesamiento y las técnicas empleadas para lograr el funcionamiento del sistema.

Palabras clave: Detección, Arañas, Brotes, Sistema Embebido, Procesamiento de imágenes.

INTRODUCCIÓN

Los bosques secos constituyen ecosistemas complejos que aportan una amplia gama de beneficios económicos, sociales y ambientales. Desde el punto de vista productivo es el hogar de muchas especies de seres vivos, para el hombre es una fuente de alimentación y de diferentes clases de suministros. El bosque también cumple la función de procesador de dióxido de carbono, amortigua los efectos del cambio climático global, protege el suelo de la erosión, recicla nutrientes, entre muchas otras. Al ser un bosque seco está expuesto a desastres como los incendios forestales, estos ocasionan graves alteraciones en el ecosistema como la destrucción y desplazamiento de los miembros del hábitat, así como también modifica la dinámica ambiental de la región. El estudio del impacto causado por los incendios es de gran interés para los expertos en temas de biología, ellos analizan y caracterizan el comportamiento que tiene la zona devastada, buscando comprender como se recupera el bioma, que especies de vegetación y animales reaparecen en la zona y cuánto tiempo tarda en iniciar la recuperación.

El grupo de investigación ambiental de la universidad Autónoma de Occidente GEADES, está desarrollando un proyecto que busca estimar el tiempo de recuperación y retorno de especies después de inducir un incendio en un área determinada en el jardín botánico de Cali, el cual entra en la denominación de bosque seco tropical. La realización de este tipo de investigación implica largas horas de trabajo de campo en la intemperie, incluyendo horas de la noche, lo que la convierte en una tarea tediosa y pesada para cualquier persona que la realice. Por esta razón el grupo GEADES busca automatizar el proceso de detección de signos de recuperación de la zona para facilitar la labor de los investigadores del grupo ambiental.

Una manera de determinar cuándo el bosque comienza a regenerarse es detectando brotes de flora por el lado de la vegetación, por parte de la fauna uno de los posibles indicadores de regreso de especies animales es la presencia de las arañas, en especial las arañas que vagan en el suelo. Por la zona y características geográficas y climáticas del terreno las especies de arañas que pueden aparecer son las Lycosidae y las Ctenidae.

Este proyecto propone para la automatización en la detección de brotes vegetales y aparición de las especies de arañas mencionadas, utilizando un sistema basado en tecnología de visión artificial, montado en un dispositivo embebido, capaz de detectar en su zona de visión la aparición de los individuos de estudio, adquiriendo una imagen del evento y permitiendo a los miembros del grupo de investigación acceder a ellas desde un punto de acceso a red.

Para poder llegar a realizar este sistema de visión artificial se deben llevar a cabo una serie de procedimientos acordes a los objetivos de detección. En primer lugar se adquieren los patrones para el reconocimiento de los brotes de flora y las arañas, encontrando las características medibles que los describen matemáticamente y que los diferencien de los demás objetos de su entorno. Partiendo de esto se plantean los algoritmos computacionales capaces de adquirir una imagen del hábitat y mediante el uso de técnicas de procesamiento digital de imágenes ajustar la imagen lo mejor posible para poder extraer los datos del individuo, de estar este en la escena. Hecho esto se aplican algoritmos que determinan si sus características concuerdan con los patrones que habían sido establecidos. Se llevan a cabo las pruebas pertinentes que determinan si el sistema diseñado cumple su función, para poder hacer la migración al dispositivo embebido. Este se compone de una cámara para la adquisición de imágenes, una unidad de procesamiento para analizar la información y un módulo para envió de datos. Cada módulo tuvo una escogencia de acuerdo a criterios definidos y su disponibilidad. Finalmente se hacen pruebas sobre el sistema ya implementado y se validan los resultados.

Como aplicación para el estudio del medio ambiente este sistema es una alternativa de solución a una necesidad de automatización para una investigación que aplica conceptos en distintos ámbitos como el análisis del comportamiento medio ambiental, el procesamiento digital de imágenes, algoritmos de identificación, uso de descriptores matemáticos, programación utilizando librerías de visión artificial, selección de plataformas hardware para implementación teniendo en cuenta la carga computacional y su compatibilidad con las librerías, la comprensión del entorno del dispositivo embebido, utilización de conceptos de servicios de red y comunicaciones. Al tener una amplia gama de aplicaciones en diferentes áreas, este proyecto puede apoyar futuras investigaciones en cualquiera de los tópicos mencionados.

1. OBJETIVOS

1.1 OBJETIVO GENERAL

Diseñar e implementar un sistema de Visión Artificial capaz de detectar una especie de Arácnido y la presencia de nuevos brotes de flora, utilizando patrones, bases de datos y algoritmos de procesamiento digital de imágenes.

1.2 OBJETIVOS ESPECÍFICOS

Estudiar la flora y tipo de Arácnido a identificar para seleccionar las características que se utilizarán para su identificación en la plataforma de implementación.

Estudiar y seleccionar los algoritmos de procesamiento de imágenes para la identificación de la especie de Arácnido y la flora.

Caracterizar y seleccionar la plataforma de implementación de acuerdo a los requerimientos de procesamiento y control del sistema.

Diseñar el sistema embebido que integre cámara, dispositivo de procesamiento, y comunicación inalámbrica.

Desarrollar pruebas que permitan validar el funcionamiento apropiado del sistema de visión artificial en la detección de Aracnidos.

2. ANTECEDENTES

Para la identificación de flora hay investigaciones como la de El Centro de Investigación en Ciencia Aplicada y Tecnología Avanzada del IPN, Unidad Querétaro (2009), a través de su estudio: **"Detección de Flores mediante el Análisis de Secuencias de Imágenes"**, presentan un método de detección de flores por medio de captura de imágenes con una video cámara y el análisis de estas utilizando distintas técnicas de identificación (segmentación, distribución de color, algoritmos, bases de datos, etc.) basadas en la forma y color de la flor, el estudio también incluye la detección de agentes polinizadores en tiempo real. El interés de la investigación es la de conocer el ciclo vital de las plantas y su impacto en el ambiente. Los resultados del estudio arrojan datos importantes sobre que algoritmos son pertinentes para la identificación de flora en ambientes naturales y en tiempo real además de detectar problemas al usar dichos algoritmos. Esta investigación está relacionada con nuestra investigación ya que entre nuestro objetivos está la detección de nuevos brotes de flora después de un incendio, lo que permite que las técnicas utilizadas tengan estrecha relación.

Miembros del **"Institute of Automation Chinese Academy of Sciences Beijing, China"** (2010), en su investigación: **"Insect Species Recognition using Sparse Representation"** (Reconocimiento de especies de insectos usando representación dispersa) plantean el reconocimiento de especies de insectos utilizando algoritmos matemáticos como el reconocimiento por representación dispersa y de aprendizaje supervisado (máquinas de soporte vectorial), buscando mejora las técnicas actuales de diferenciación de insectos que usan métodos más genéricos. Para la identificación de las especies generaron una base de datos con imágenes de las partes de los insectos para realizar los patrones de entrenamiento del sistema. Los resultados de la investigación permiten conocer cómo se aplican las bases de datos para generar patrones de entrenamiento, de la importancia de utilizar un gran número imágenes diferentes para generar las bases de datos ya que de lo contrario los resultados no son los esperados, también es importante conocer cómo se realiza la implementación y aplicación de los algoritmos mencionados. Este estudio tiene y aporta valiosos datos a nuestra investigación y desarrollo del proyecto ya que entre nuestro objetivos está la identificación de una especie de Arácnido en un ambiente determinado.

Para el caso puntual de la identificación de arañas, Do, Harp y Norris (1999), en su estudio: **"A test of a pattern recognition system for identification of spiders"** (Una prueba de un sistema de reconocimiento de patrones para identificación de arañas) con el fin de facilitar el trabajo de identificación de especies de arañas en vista de la disminución de taxonomistas, proponen un sistema de reconocimiento de patrones utilizando redes neuronales artificiales. Varias redes neuronales

fueron entrenadas para la identificación de especies de arañas usando únicamente fotos de los órganos genitales de la hembra, de las cuales la información clave fue extraída utilizando transformada de Wavelet. Se realizó la prueba para la araña lobo, se realizaron pruebas con tres redes neuronales de diferentes tamaños. Los resultados de la investigación arrojaron que la red de mayor tamaño tuvo 100% de acierto al momento de hacer la identificación, además de tener en cuenta que no utilizaron una base de datos para el entrenamiento de la red, muy grande. La información sobre la adquisición de imágenes y su procesamiento son de gran utilidad para nuestro proyecto ya que presentan varios puntos en común, los algoritmos presentados y la implementación de las redes son muy importantes para tener en cuenta a la hora de hacer nuestra propia investigación y desarrollo del proyecto.

3. DESARROLLO CONCEPTUAL

3.1 ARAÑAS EN EL VALLE DEL CAUCA

Este proyecto tiene como planteamiento la detección e identificación de una especie de araña con fines de adquisición de datos para estudios medioambientales después de un incendio. En este capítulo se describe el procedimiento realizado para la escogencia de la familia y especie de araña a ser identificada por el sistema de visión artificial propuesto.

3.1.1 Selección de la Especie. Al estar la zona de detección ubicada a nivel del suelo, la araña a identificar debía ser entonces del tipo Errante (vagan por el suelo). Lo que determina, que las familias más adecuadas para ser sujetos de detección son las *Lycosidae* (Figura 1) y las *Ctenidae* (Figura 2).

Figura 1. Araña Lycosidae

Fuente: Lycosidae Male Alopecosa fabrilis, Suecia, Junio 2008, consultado 17 noviembre 2012, disponible en Internet: http://www.eurospiders.com/family_Lycosidae.htm.

Figura 2. Araña Ctenidae

Fuente: Large wandering spiders in Amazonian forests, consultado 18 noviembre de 2012, disponible en Internet: http://www.wandering-spiders.net/ctenidae/introduction/

En el artículo *"Cambios en la Comunidad de Araneae Durante la Sucesión Postfuego en Matorrales Mediterráneos de Montaña"* (Urones& Majadas), se muestra como en la etapa descolonización del primer año después de la devastación por el fuego todos los ejemplares encontrados estaban sobre el suelo y que la familia *Lycosidae* tiene preferencia en aparecer en la etapa inicial después del evento (Tabla 1), claro está que este estudio no se realizó en la ciudad de Cali bajo las mismas condiciones climáticas y geográficas, pero es un buen indicio de que es probable encontrarlas en una región de pos fuego.

En biología, específicamente en categorías taxonómicas, es virtualmente imposible dar un número estimado de probabilidad de aparición, pues se sabe muy poco, sería bastante cuestionable tan solo comentar si la ocurrencia se podría medir con una cifra. A pesar de esto haciendo uso de software computacional que con base en variables climáticas, altitudinales, ecológicas y modelo de entropía, puede "predecir" el hábitat más "adecuado" para una especie; en este caso de Lycosidae y Ctenidae, estas son familias con cientos de especies, por lo que el modelo de entropía no funciona muy bien dado que cada especie tiene una altura, un clima y un tipo de hábitat ecológico que prefiere, y no se sabe a ciencia cierta que especies habitan en la región.

Tabla 1. Tabla de aparición de arañas pos fuego.

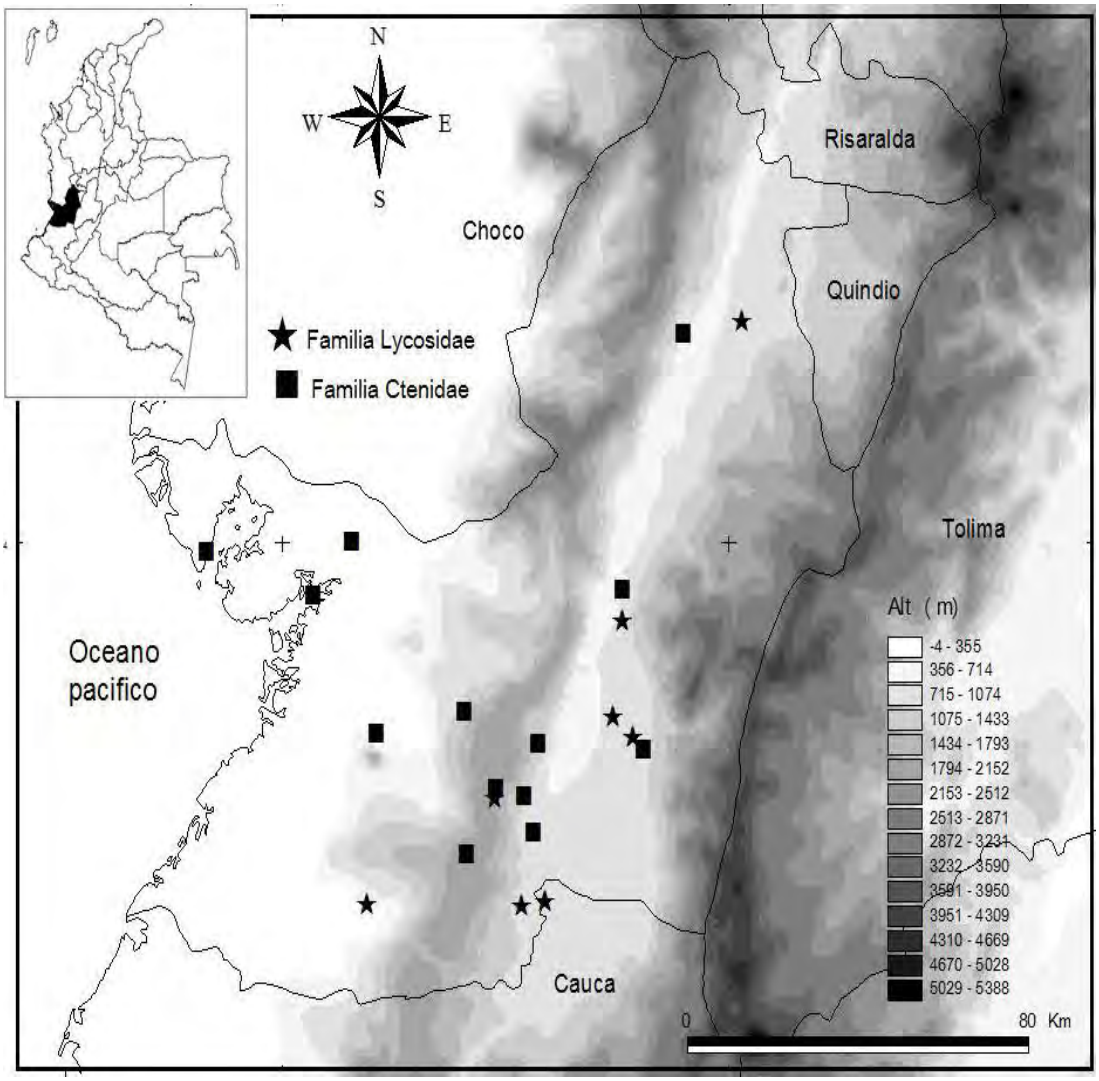
Número de ejemplares (n), de especies (s) y abundancia (A%) para cada familia de arañas recogidas en cada etapa de la sucesión postfuego en los piornales de montaña.

FAMILIAS	Inicial (0-1 a)			Temprana (2-6 a)			Media (7-9 a)			Tardía (10+ a)		
	n	s	A%	n	s	A%	n	s	A%	n	s	A%
Agelenidae	–	–	–	131	1	44,1	69	1	18,90	46	2	8,66
Araneidae	3	2	6,99	64	3	21,55	123	3	33,70	70	7	13,18
Clubionidae	–	–	–	1	1	0,34	1	1	0,27	–	–	–
Dictynidae	–	–	–	–	–	–	1	1	0,27	–	–	–
Eresidae	–	–	–	–	–	–	–	–	–	1	1	0,19
Gnaphosidae	–	–	–	9	4	3,03	8	3	2,19	1	1	0,19
Linyphiidae	4	2	9,31	50	3	16,83	46	5	12,60	143	6	26,93
Liocranidae	–	–	–	–	–	–	–	–	–	1	1	0,19
Lycosidae	21	3	48,86	8	5	2,69	8	3	2,19	50	2	9,42
Nemesiidae	1	1	2,33	4	1	1,35	37	1	10,14	6	1	1,51
Oxyopidae	1	1	2,33	4	1	1,68	12	2	3,29	16	2	3,01
Philodromidae	4	2	9,32	5	2	2,36	18	3	4,93	77	2	14,51
Salticidae	6	2	13,96	9	2	3,37	6	2	1,64	10	6	1,88
Tetragnathidae	–	–	–	–	–	–	–	–	–	1	1	0,19
Theridiidae	1	1	2,33	8	1	2,69	32	2	8,77	87	3	16,35
Thomisidae	2	1	4,66	1	1	0,34	3	1	0,82	22	1	4,14
Zodariidae	–	–	–	3	1	1,01	1	1	0,27	–	–	–
Total	**43**	**15**	**100**	**297**	**26**	**100**	**365**	**29**	**100**	**531**	**37**	**100**

Fuente: URONES, Carmen. MAJADAS, Arturo**.** Articulo Cambios en la Comunidad de Araneae Durante la Sucesión Postfuego en Matorrales Mediterráneos de Montaña" (Urones& Majadas), Revista Ibérica de Aracnología, Vol. 5, 31-VII-2002, Sección: Artículos y Notas. P: 19–28.

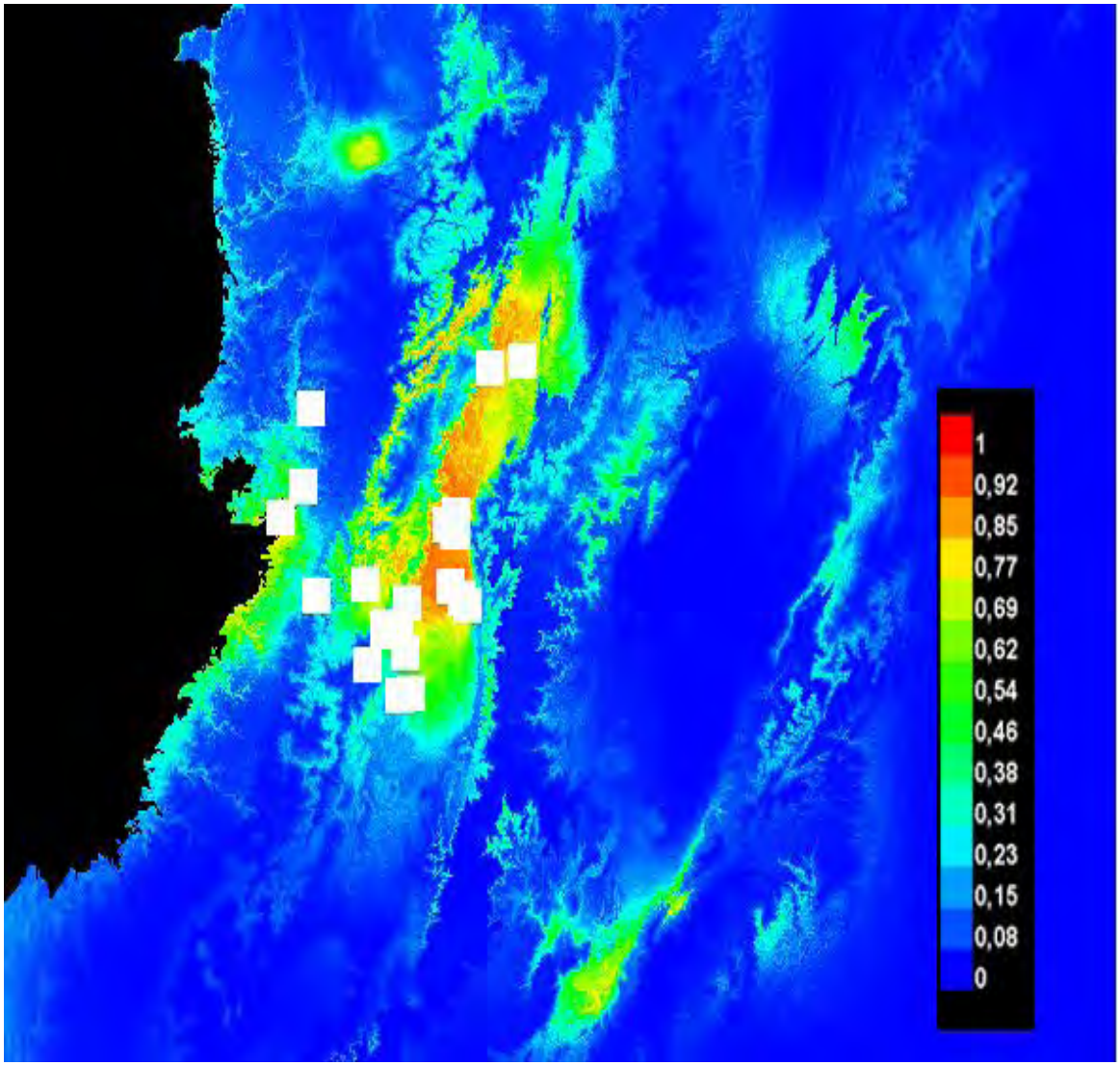
A partir de una base de datos de localidades de las arañas de ambas familias (Lycosidae y Ctenidae) que se encuentran depositadas en el Museo de Entomología de la Universidad del Valle (MUSENUV) y utilizando el programa informático Maxent, se generó un mapa con la ubicación de las localidades de las arañas según las coordenadas de la base de datos (Figura 3) y otro mapa con probabilidades de ocurrencia de Lycosidae y Ctenidae (Figura 4).

Figura 3. Mapa distribución de localidades



Fuente: MONTOYA, James, Departamento de Biología sección de entomología de la Universidad de Valle, diciembre 05 de 2012.

Figura 4. Mapa de probabilidad de ocurrencia



Fuente: MONTOYA, James, Departamento de Biología sección de entomología de la Universidad de Valle, diciembre 05 de 2012.

Como se puede apreciar en los mapas en donde está situada la ciudad de Cali, en general, las probabilidades de hallar una de las dos familias de arañas, es superior al 0,38 o 38%, lo que se considera en biología para este tipo de especies, altamente probable.

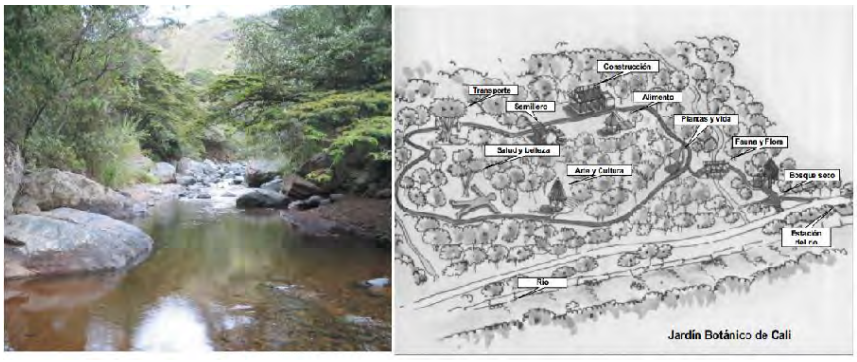
Una vez caracterizada y seleccionada la especie a detectar y teniendo en cuenta las limitaciones del sistema, se determinaron las restricciones en las que el sistema puede funcionar de forma óptima para la detección, las cuales son: el tamaño de la araña, el cual debe ser mínimo de 2cm y la luz presente en el medio.

Durante el día el sistema puede detectar la araña sobre objetos como hojas, tallos y piedras, durante la noche el sistema puede detectar la araña pero sin objetos que puedan cambiar la forma del arácnido u obstaculicen la detección, para los brotes de flora el sistema puede detectar tanto de día como de noche.

3.2 FLORA EN EL JARDÍN BOTÁNICO DE CALI

El jardín botánico de Cali (Figura 5) es un área protegida de bosque seco tropical de once Hectáreas ubicadas en la cuenca del río Cali, creado con la misión de promover la conservación de la biodiversidad del Valle del Cauca mediante programas de investigación, educación ambiental, conservación y horticultura con el propósito de mejorar la calidad de vida de las comunidades en armonía con la naturaleza.

Figura 5. Jardín Botánico de Cali



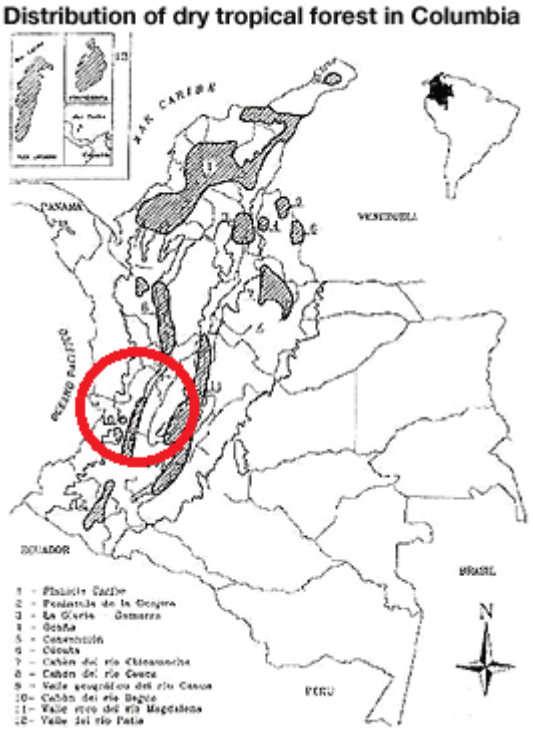
Fuente: OREJUELA G, Jorge E, Ph.D Jefe del departamento de ciencia Ambientales, Universidad Autónoma de Occidente, el hombre y la maquina No 22, Julio de 2014. Disponible en internet: http://bdigital.uao.edu.co/bitstream/10614/207/1/T0003287.pdf

El Bosque seco Tropical (Bs-T) se define como aquella formación vegetal que presenta una cobertura boscosa continua y que se distribuye entre los 0-1000 m

de altitud; presenta temperaturas superiores a los 24º C (piso térmico cálido) y precipitaciones entre los 700 y 2000 mm anuales, con uno o dos periodos marcados de sequía al año (Espinal 1985; Murphy & Lugo 1986, IAVH 1997).

En Colombia este bosque se desarrolla en lugares con precipitación que fluctúa entre 789 mm (Isla de Tierra Bomba, Bolívar) y los 1800 mm (pie de monte de la cordillera central Valle del Cauca). La temperatura media anual es superior a los 25º C, alcanzando temperaturas máximas de 38º C (IAVH 1995, 1997; CVC 1994). En la figura 6 se muestra la distribución del bosque seco en Colombia y encerrado en rojo el Departamento del Valle del Cauca.

Figura 6. Mapa de Distribución Bosque Seco en Colombia



Fuente: Tropical Forest and Cotton-Top Tamarin Hábitat, consultado 25 noviembre de 2012, disponible en Internet: http://proyectotiti.com/english/tropical-forests-of-colombia.htm

En lo que se refiere al tipo de vegetación que se puede encontrar dentro del jardín botánico, se puede realizar una distinción por su altura. Entre los arboles grandes que son el “techo” del bosque, se destacan higuerones, chambimbes, cedrillos, jiguas, gualandayes, chiminangos, guásimos, arrayanes, jaguas y carboneros.

Luego aparece la franja arbustiva que está entre los 2 y 5 metros, entre las especies arbustivas y rastreras se destacan las acacias chirlobirlo, cují o trupillo, clavellinos, dorancés o martingalvis, carboneros y la dormidera. El bosque también es rico en El bosque es rico en bejucos y enredaderas (Publicación trimestral de la Fundación Jardín Botánico de Cali - Número 3 - Año 2004). Por último se puede incluir la hierba común o pasto, que varía según la cantidad de precipitación de agua y la cercanía con la fuente hidrográfica. Comúnmente en los bosques secos tropicales se aprecia la pérdida del follaje como una adaptación fisiológica de las plantas del bosque al déficit de agua.

El interés de estudio de este proyecto se encuentra en la detección a nivel del suelo, este se caracteriza como ya se mencionó por tener apariciones esporádicas de vegetación, con predominio de terreno seco con una tonalidad oscura, después de un incendio puede oscurecerse aún más como se puede observar dentro del circulo amarillo en la Figura 7.

Figura 7. Terreno en Jardín Botanico con Quemado

Así mismo se puede ver el color que tiene la hierba que se encuentra a nivel del suelo, esto es lo que se tiene en cuenta a la hora de detectar los nuevos brotes de flora que se plantean dentro de los objetivos del proyecto.

3.3 ALGORITMOS DE PROCESAMIENTO

Todo sistema de visión artificial (SVA) tiene como propósito la “comprensión” de las características de una imagen para facilitar y automatizar tareas que de otro modo tendrían que ser realizadas por un humano. Para entender el concepto de visión artificial se puede utilizar una comparación con el sistema de visión humano (SVH).

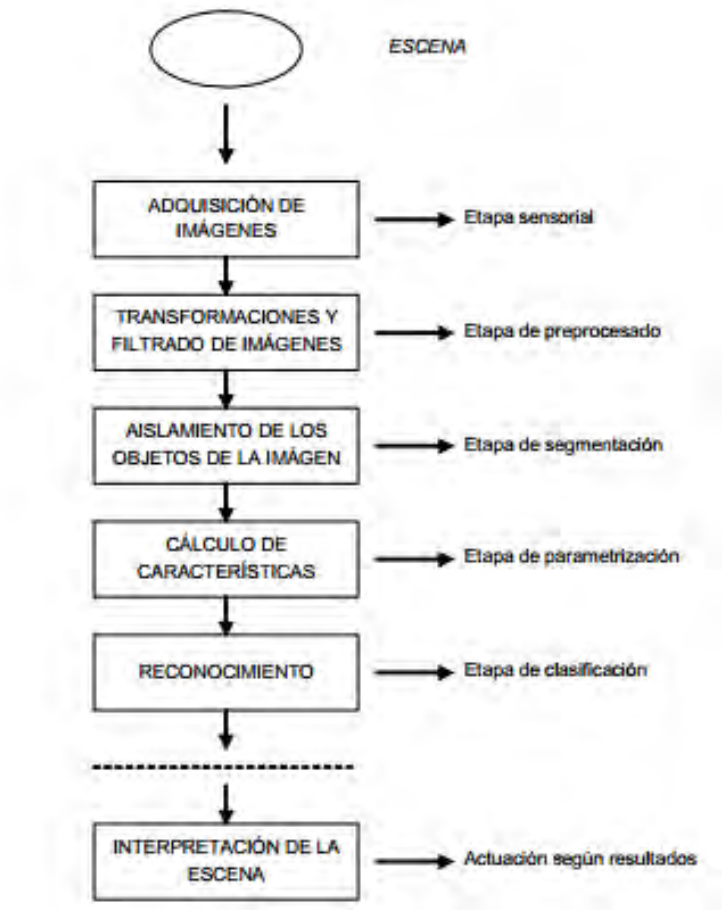
El SVH interpreta las imágenes basándose en reconocimiento por experiencia, por su parte el SVA utiliza instrucciones programadas por un humano para extraer características físicas y convertirlas en una cantidad medible a la cual se le puedan aplicar métodos matemáticos y lógicos para interpretarlos siguiendo pasos sucesivos, a este proceso se le conoce como algoritmo.

Este capítulo presenta las etapas desarrolladas para construir el programa que contiene los algoritmos utilizados para cumplir con el objetivo de detectar y reconocer una araña de las familias *Lycosidae o Ctenidae.*

3.3.1 Estructura general. La estructura general de un sistema de visión artificial (Figura 8) está compuesta por, una etapa sensorial que es la encargada de la captura de la imagen, una etapa de preprocesamiento cuyo objetivo es mejorar la imagen de forma que el objeto final tenga mayores posibilidades de éxito en la identificación, una etapa de segmentación la cual consiste en diferenciar los diversos objetos del fondo de la imagen, una etapa de parametrización la cual realiza los cálculos matemáticos e identificación de características únicas del objeto, para construir los descriptores que permitirán la identificación del objeto, por último la etapa de reconocimiento, evalúa cada uno de los descriptores para determinar si el objeto es el deseado y dar una interpretación de la escena para realizar una acción.

Las etapas de preprocesado y segmentación son las más importantes de todo el sistema de visión artificial, ya que de estas depende el óptimo reconocimiento del objeto y donde se presenta el mayor gasto computacional por los algoritmos de procesamiento.

Figura 8. Diagrama de bloques de un sistema de visión artificial



Fuente: MARCOS GONZÁLEZ, Ana, integrantes del grupo de investigación EDMANS. Universidad de La Rioja, 2006. p. 16.

3.3.2 Preprocesado y segmentación. Una vez se captura la imagen se debe realizar el preprocesado de la imagen para reducir o eliminar el ruido presente por la falta de iluminación y calidad de la cámara. El objetivo del preprocesamiento es mejorar la imagen de forma que el objeto final tenga mayores posibilidades de éxito en la identificación. Para conseguir esto se emplean técnicas de: realce de contornos, eliminación de ruido, aumento de contraste, brillo y corrección gamma.

Realce de Contornos: El propósito de las técnicas de realce, es mejorar la apariencia de la imagen, la selección de los métodos apropiados y la elección de los parámetros, dependen de la calidad de la imagen original y de la aplicación.

Las técnicas de realce de contornos son conocidos como filtro pasa altos, este tipo de filtros pretenden aislar los componentes de alta frecuencia en una imagen lo que significa realzar los pequeños detalles en una imagen para mejorar la nitidez.

Estos filtros también pueden ser utilizados para reforzar los bordes presentes en una imagen sin importar su orientación.

Figura 9. Mascaras de filtros pasa altos.

-1	-1	-1
-1	9	-1
-1	-1	-1

(a)

0	-1	0
-1	5	-1
0	-1	0

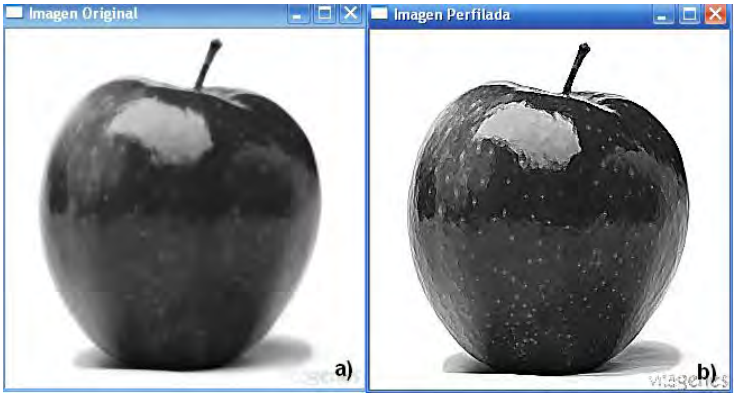
(b)

1	-2	1
-2	5	-2
1	-2	1

(c)

Fuente: MEDINA Rubén, BELLERA Jesús. Grupo de Ingeniería Biomédica de la ULA. Universidad de Los Andes. Bases del Procesamiento de Imágenes Médicas. Venezuela. P.14

Figura 10. Procesamiento filtro pasa alto



La figura 10 (b), presenta una mejor nitidez y perfilado que la imagen original (a), la máscara va recorriendo la matriz de la imagen, si el píxel central posee un valor de brillo muy diferente al de sus vecinos inmediatos, entonces el efecto de estos últimos es despreciable y el valor de salida es una versión acentuada del valor original del píxel central. Por el contrario, si los valores de brillo de los píxeles vecinos son suficientemente grandes para contrarrestar el peso del píxel del

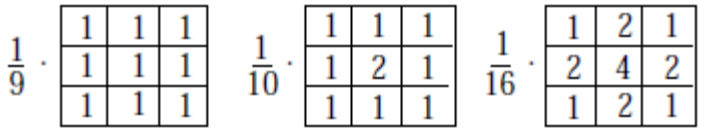
centro, entonces el resultado final se basa más en un promedio de los píxeles involucrados.

Eliminación de Ruido: Los operadores de suavizado se utilizan en la eliminación de ruido, para esto se emplean filtros pasa bajos y filtros promediadores.

El filtro de paso bajo se utiliza para restaurar errores aleatorios que pueden presentarse en los niveles de brillo de la imagen, producto de un defecto en la adquisición o recepción de los datos.

Filtro de la mediana, se basa en sustituir el valor de brillo del píxel central del núcleo por la mediana de todos los valores de brillo de los píxeles que forman dicho núcleo. Se emplea para eliminar valores anómalos aislados, producto de ruidos aleatorios en la adquisición de los datos.

Figura 11. **Mascaras de filtros pasa bajos**



Fuente: MEDINA Rubén, BELLERA Jesús. Grupo de Ingeniería Biomédica de la ULA. Universidad de Los Andes. Bases del Procesamiento de Imágenes Médicas. Venezuela. P.12

La implementación del filtro Gaussiano 3x3 y 5x5 (Figura 12), es ideal para quitar ruido gaussiano, pero a medida que aumenta el orden del kernel la imagen se desenfoca además de presentar rasgos de ruido en la imagen final.

El filtro Mediana de orden 3x3 y 5x5 (Figura 13), es útil para reducir el ruido impulsivo, además de mejorar la textura del fondo de la imagen lo que permite tener un mejor contraste entre el objeto y el fondo de la imagen, lo que permitirá una mejor segmentación en etapas posteriores.

Figura 12. Filtro gaussiano de 3x3 y 5x5

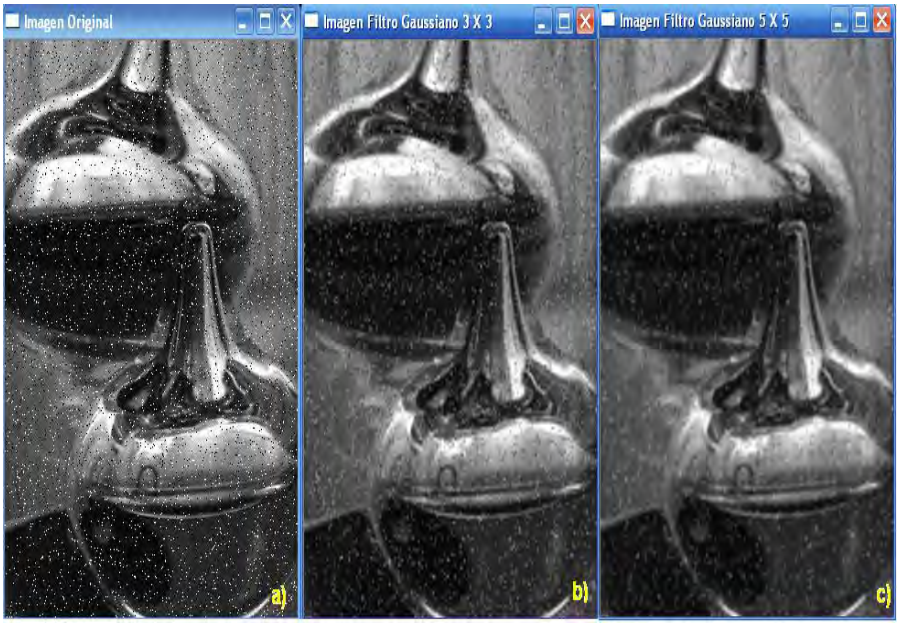
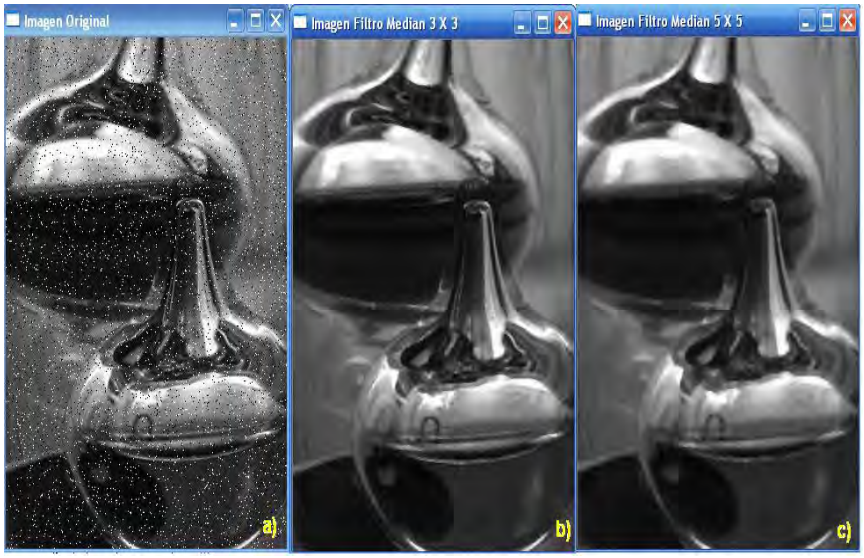


Figura 13. Filtro mediana de 3x3 y 5x5



Aumento de Contraste y Brillo: El contraste es un término usado para denotar el grado de diferencia entre los componentes más oscuros y los más claros en una imagen.

Una imagen tiene un buen contraste si se compone de una amplia gama de valores desde el negro al blanco. Durante una variación en el contraste, el valor de cada píxel es modificado por un valor de contraste, que sirve para redistribuir las intensidades sobre una gama más estrecha o más amplia. Al modificar el contraste de las imágenes, a mayor contraste la imagen tiende a convertirse a imagen a blanco y negro, mientras que cuando se disminuye, esta tiende a convertirse a escala de grises (Figura 14).

El brillo se define como la intensidad de la luz en cada píxel de una imagen, al visualizar una imagen con variaciones en el brillo, esta se verá mucho más clara o más oscura de acuerdo a qué tanto brillo tendrá esta. Cuando se varía el brillo en una imagen todos los píxeles de la imagen modifican su luminosidad en igual cantidad, Cuando se incrementa el brillo, el valor de cada píxel se acerca más a 255 (blanco). Cuando se disminuye, el valor de cada píxel se reduce más cerca del 0 (negro) (Figura 15).

Figura 14. Aumento de contraste

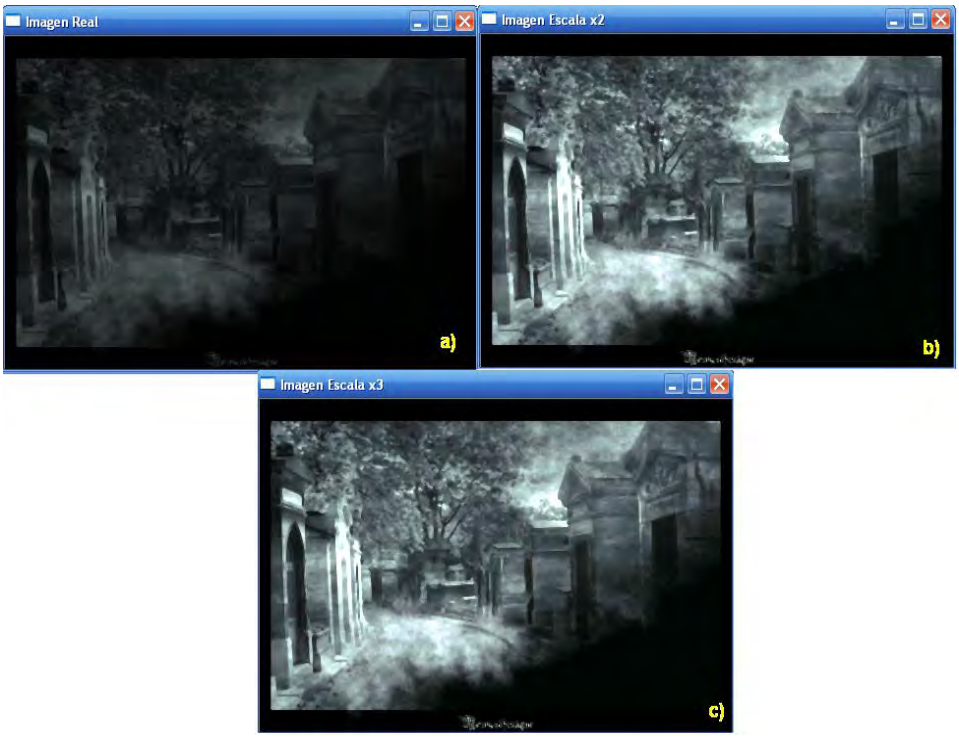
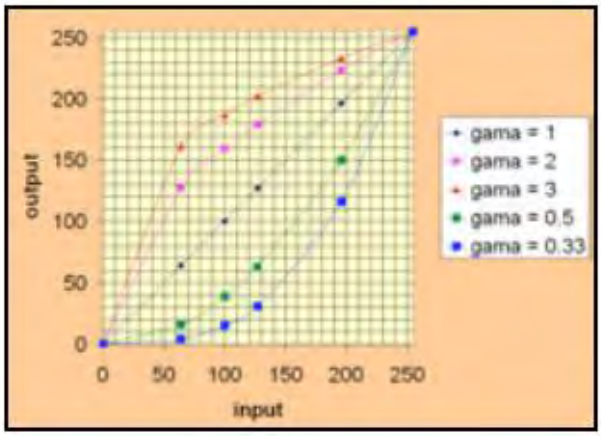


Figura 15. Aumento de brillo

Corrección Gamma: Se utiliza para ajustar los valores de intensidad de una imagen es una combinación del brillo y el contraste con el objetivo de compensar las variaciones de intensidades de las imágenes (Figura 16).

Figura 16. Concepto Corrección Gamma



Para gamma = 1 no hay ninguna corrección. Para gamma > 1 hay una gran corrección en el contraste para valores pequeños del color de entrada, mientras que una pequeña corrección en el contraste para valores grandes. El brillo aumenta más para valores intermedios del color de entrada. Para gamma < 1 hay una pequeña corrección en el contraste para valores pequeños del color de entrada mientras que una gran corrección en el contraste para valores grandes. El brillo disminuye más para valores intermedios del color de entrada.

En la figura 17 se muestra el resultado de implementar la corrección gamma, imagen original (a), gamma = 0.5 (b), gamma = 2 (c).

Figura 17. Corrección Gamma

Segmentación

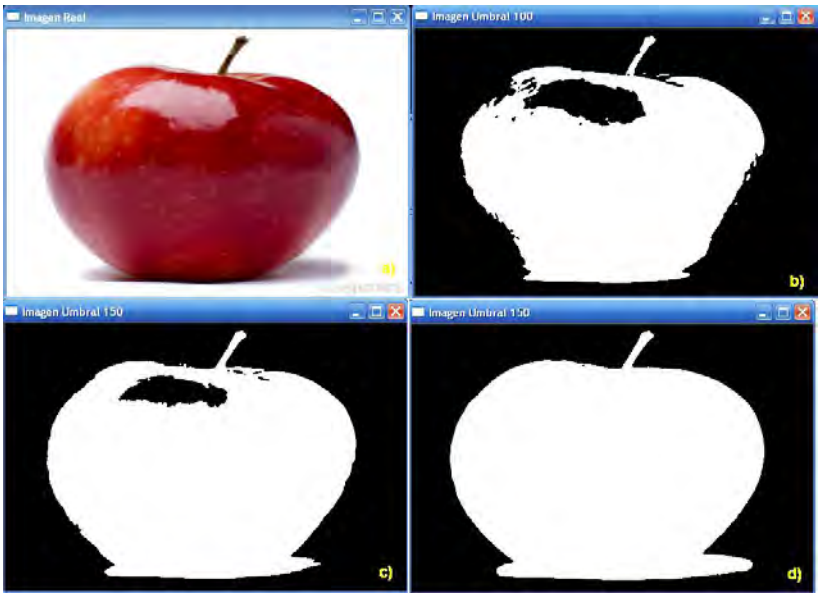
El objetivo de la segmentación es subdividir la imagen en regiones también llamados clases o subconjuntos. Es decir, consiste en diferenciar los diversos objetos del fondo de la imagen. Al final de la etapa de segmentación, se tienen que conocer perfectamente los objetos que hay, para extraer las características propias de cada uno de ellos.

La segmentación basada en umbralización se usada para identificar y separar objetos del fondo, en base a la distribución de los niveles de gris o la textura de los objetos en las imágenes. Muchas técnicas de umbralización se basan en la estadística del histograma de los niveles de gris o en la matriz de co-ocurrencia de una imagen. La determinación de los umbrales puede ser mediante procedimientos paramétricos o no paramétricos.

En los procedimientos paramétricos, la distribución de los niveles de gris de un objeto permite obtener el umbral a partir de procedimientos estadísticos. En los procedimientos no paramétricos, los umbrales se obtienen de una forma óptima, de acuerdo a algún criterio.

Threshold: Este un método basado en regiones no paramétrico se emplea para binarizar la imagen a partir de un valor de umbral (Figura 18).

Figura 18. Procesamiento por Threshold



El valor de umbral se obtiene de forma óptica a partir del histograma, a medida que el valor de umbral aumenta el objeto se separa del fondo pero empiezan aparecer valores no deseados o imágenes incompletas del objeto real, imagen (b) y (c).

Método de Otsu: El algoritmo consiste en obtener para todos los umbrales posibles, un valor denominado varianza entre clases permite medir el grado de diferencia entre los píxeles y un rango establecido. El umbral que maximice esta varianza entre clases es el óptimo para la umbralización. En la figura (19) se muestra el resultado del método Otsu a partir del histograma de la figura (20).

Figura 19. Implementación método Otsu

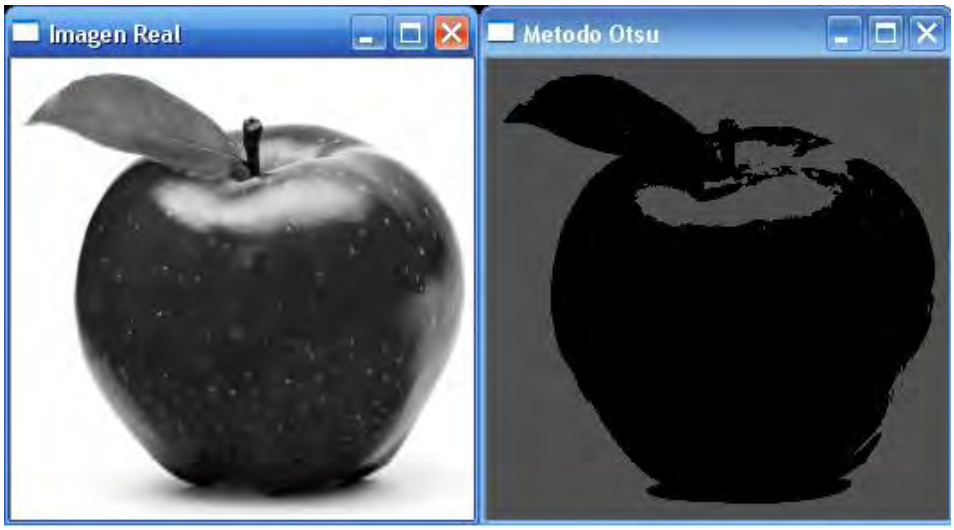
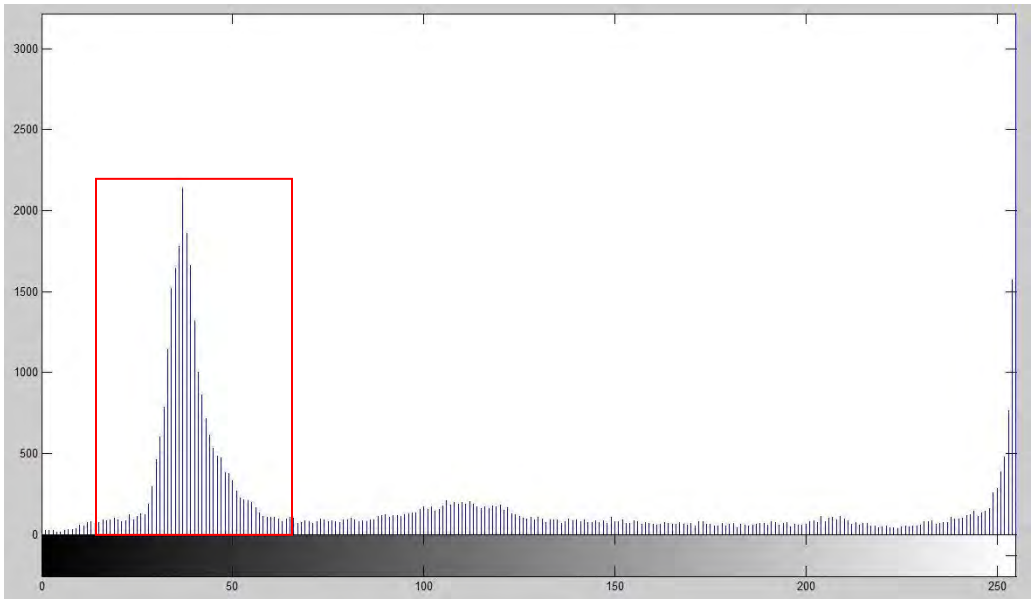


Figura 20. Histograma para implementación método Otsu



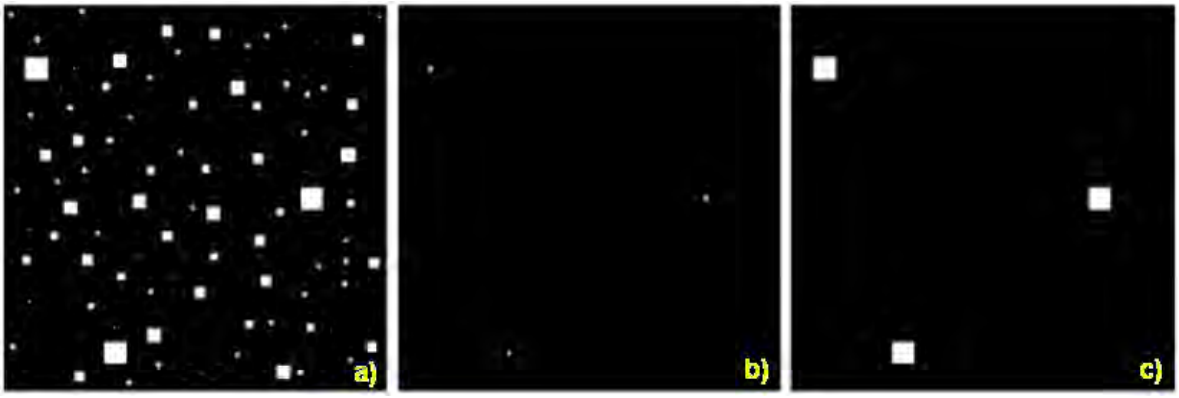
Operaciones Morfológicas

La morfología matemática se basa en la geometría y forma, simplifican imágenes y conservan las principales característica de forma de los objetos. También se usan para extraer componentes de imágenes útiles para la representación y descripción. Las operaciones primarias morfológicas son la erosión y la dilatación. A partir de estas se pueden componer las operaciones de apertura y clausura.

De forma práctica, La erosión permite eliminar detalles irrelevantes del objeto imagen (b), para volver a tener el objeto del tamaño original se debe emplear la dilatación con el mismo elemento estructural, imagen (c), esto significa que la dilatación no restaura los elementos erosionados, figura (21).

La Apertura, generalmente suaviza el contorno de un elemento, rompe uniones angostas y elimina salientes finas. El Cierre, también tiende a suavizar contornos, pero a diferencia de la anterior, une cortes en partes angostas y finas, elimina pequeños huecos y llena baches en los contornos.

Figura 21. Operación de Erosión y Dilatación



3.3.3 Detección de Contornos. La detección de contornos es una de las etapas del proceso de la segmentación, un borde en una imagen es un límite en el cual ocurren cambios significativos en algún parámetro físico como intensidad de color o cambio en la textura.

Para la detección de contornos se emplean diferentes técnicas entres las más usadas están, operador de *Sobel y prewitt*, operador de *Roberts* y algoritmo de *canny*.

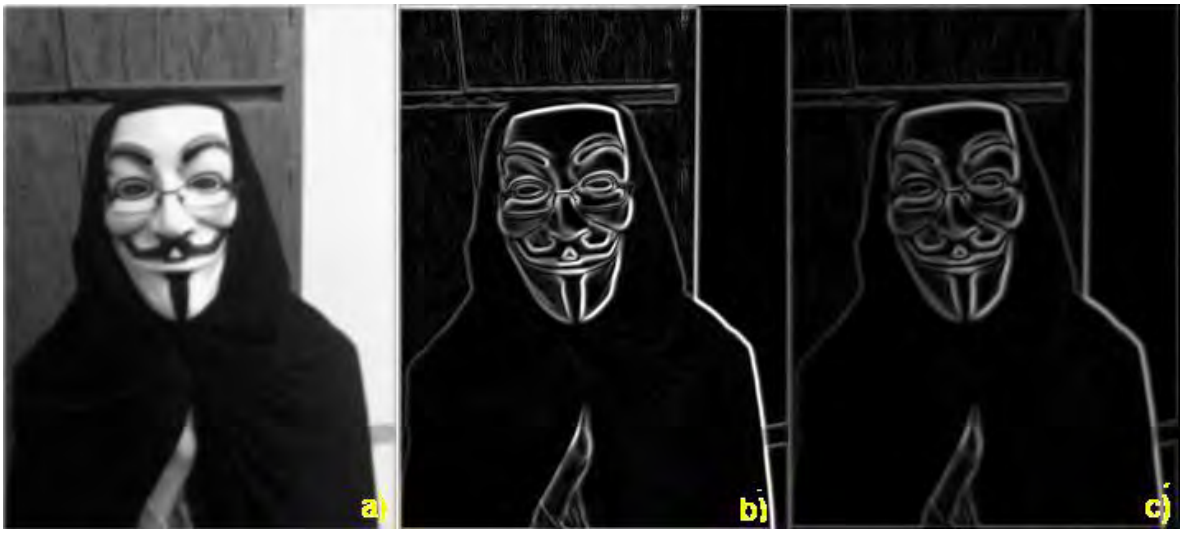
Operador de Sobel y Prewitt.

Este operador calcula el gradiente de intensidad de brillo de cada píxel dando la dirección del mayor incremento posible (de negro a blanco), el resultado muestra que tan abruptamente cambia una imagen en cada punto analizado, en la figura (22) la imagen (b) es el resultado del operador sobel y la figura (c) del operador prewitt.

La ventaja de este operador es que tiene poco gasto computacional debido a sus dos únicas matrices para la detección de bordes verticales y horizontales, pero no detecta todos los bordes de la imagen lo que no es útil para la detección del objeto.

$$\mathbf{G}_x = \begin{bmatrix} -1 & 0 & +1 \\ -2 & 0 & +2 \\ -1 & 0 & +1 \end{bmatrix} * \mathbf{A} \quad \text{and} \quad \mathbf{G}_y = \begin{bmatrix} +1 & +2 & +1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} * \mathbf{A}$$

Figura 22. Operación del operador Sobel y Prewitt



Operador Roberts.

Este tipo de operador presenta buenos resultados ante bordes diagonales y localización de los mismos, pero presenta extrema sensibilidad al ruido por lo tanto tiene pobres cualidades de detección.

Figura 23. Operador Roberts

Algoritmo de Canny.

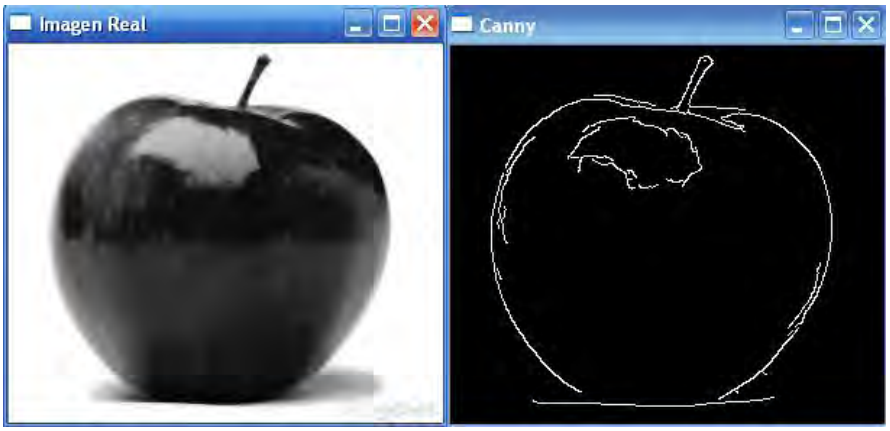
Es un operador que utiliza un algoritmo de múltiples etapas para detectar una amplia gama de bordes en la imagen, este algoritmo está considerado como uno de los mejores métodos de detección de contornos mediante el empleo de máscaras de convolución y basado en la primera derivada.

Este método de detección de bordes, está basado en tres criterios, estos son: Evitar la eliminación de bordes importantes y no suministrar falsos bordes, la distancia entre la posición real y la localizada del borde se debe minimizar y no identificar múltiples píxeles de contornos donde sólo exista un único contorno.

Esta técnica, que se caracteriza por estar optimizada para la detección de bordes diferenciales, consta de tres etapas principales: filtrado, decisión inicial, e histéresis. La primera etapa consiste en un filtrado de convolución de la derivada primera de una función gaussiana normalizada discreta sobre la imagen, realizada

en dos direcciones horizontal y vertical. La segunda etapa realiza una primera toma de decisiones sobre los posibles bordes de la imagen, mediante el cálculo de los picos de las imágenes obtenidas en la primera etapa. La última etapa de procesamiento realiza una optimización de la decisión llevada a cabo en la etapa anterior, mediante la aplicación de una función de histéresis sobre la imagen. Esta función se basa en la definición de dos umbrales, TL y TH, tales que TL < TH. La implementación del filtro se muestra en la figura (24).

Figura 24. Algoritmo de Canny



3.3.4 Color. El color es un aspecto importante en el tratamiento de imágenes ya que es un componente que aporta información valiosa sobre las características de una escena en una imagen y sobre los objetos presentes en esta. Muchos de los análisis y operaciones que se realizan en el procesamiento digital de imágenes, como algunos de los vistos en puntos anteriores de este mismo capítulo, se basan en las componentes acromáticas de la imagen, es decir, no se tienen en cuenta los colores sino la cantidad de luz (intensidad) que emiten, se suele representar con niveles de gris, haciendo referencia a una medida escalar de intensidad que va del negro al blanco, pasando por los grises.

Para la descripción de los componentes de color de la imagen o la fuente cromática de luz, se emplean tres magnitudes básicas:

- Radiancia: Cantidad total de energía que sale de la fuente luminosa (W).
- Luminancia: Cantidad de energía que un observador percibe procedente de una fuente luminosa (lm).
- Brillo: Descriptor subjetivo, incluye la noción acromática de la intensidad.

Además de estas características para la descripción del color, hay otras características que sirven para distinguir un color de otro, estas son además del ya mencionado brillo:

- Tono: Atributo asociado con la longitud de onda dominante en una mezcla de ondas luminosas.
- Saturación: Cantidad de luz blanca mezclada con un tono.

Juntos tono y saturación constituyen la cromaticidad, en resumen, un color se puede caracterizar por su brillo y cromaticidad. Sabiendo esto se puede procesar una imagen ya sea aislando los componentes o utilizarlos como una herramienta descriptora.

A la hora de hacer el tratamiento de las imágenes se suelen aprovechar lo que se conoce como modelos de color, que buscan facilitar la especificación de los colores de una forma normalizada. El modelo es la especificación de un sistema de coordenadas tridimensional en el que cada color es representado por un único punto.

Existen gran cantidad de modelos de representación de color que se utilizan en distintas aplicaciones, ya sea para la manipulación del color o los orientados al hardware (monitores, televisión, etc.). Entre los más destacados para el tratamiento de imágenes en programas con librerías de visión artificial se encuentran:

- RGB: De la sigas en ingles Red, Green, Blue (Rojo, Verde, Azul), consiste en la representación de un color mediante la mezcla por adición de los tres colores primarios. Las imágenes del modelo de color RGB consisten en tres planos de imagen independientes, uno por cada color primario, en cada uno el valor de cada píxel suele ir de 0 a 255, según la cantidad de cada color primario en este píxel.

- HSV: De la siglas en ingles Hue, Saturation, Value (Tono, Saturación, Valor), en este modelo, el tono se representa como un grado de ángulo cuyos valores posibles van de 0 a 360° o de 0 a 1, la saturación se representa entre valores de 0 a 1 y el valor viene siendo prácticamente la componente acromática de la imagen, va de 0 a 1.

- YCbCr: Es una familia de espacios de colores usada en sistemas de transmisión de vídeo y fotografía digital. La Y es la componente de luminancia, Cb es la componente de color azul (B-Y) y Cr es la componente de color rojo (R-Y).

Manipulando estos modelos de color se puede hacer un análisis tanto en la parte acromática como en la de color, expandiendo las opciones a la hora de procesar una imagen, por ejemplo en caso de que una imagen en su componente acromática presente información difusa al momento de llevar a cabo una segmentación de objetos presentes en una escena para una posterior identificación, se puede considerar la utilización del color como una herramienta de segmentación.

3.3.5 Descriptores. Una vez realizada la segmentación de la imagen en regiones, los píxeles resultantes de la segmentación deben ser descritos en una forma adecuada para un posterior reconocimiento e identificación. Esto significa seleccionar características descriptoras del objeto a evaluar, que lo diferencien de otros objetos comunes en su entorno.

Los esquemas de representación de formas deben de tener ciertas propiedades deseables:

- **Unicidad:** cada objeto debe tener una única representación.
- **Invariancia** frente a transformaciones geométricas, como traslaciones, rotaciones, cambios de escala y reflexiones.
- **Sensibilidad** o capacidad para diferenciar objetos casi iguales.
- **Abstracción** del detalle o capacidad para representar los rasgos característicos básicos de los objetos y abstraer los detalles.

Existen dos formas de representación:

Esquemas de representación externa, usan el contorno de los objetos y sus rasgos característicos, como son los códigos de cadena, los descriptores de Fourier y las aproximaciones poligonales.

E*squemas de representación interna*, que describen la región ocupada por el objeto en la imagen binaria, como son el área, los momentos y los esqueletos.

Códigos de Cadena: El código de cadena es una representación invariante frente a *traslaciones.* Esta propiedad facilita la comparación de objetos. A partir del código de cadena se pueden obtener ciertas características del contorno, como el perímetro, el área del objeto y los descriptores de Fourier.

Descriptores geométricos: Permiten obtener a partir de la forma geométrica los contornos de las regiones (objetos), son valoraciones numéricas que permiten identificar y reconocer los objetos en una imagen. Los principales descriptores geométricos son: perímetro, diámetro y excentricidad.

Descriptores de Regiones: Estos descriptores caracterizan el interior de la forma del contorno, suelen ser más robustos ante ruido (variaciones pequeñas del contorno) y ante transformaciones geométricas. En el cuadro 1 se presentan los principales descriptores basados en regiones:

Cuadro 1. Descriptores de regiones

Descriptor	Definición	Si es circulo	Si es cuadrado
Factor de Forma	$\frac{4\pi * Área}{Perimetro^2}$	1	$\frac{\pi}{4}$
Redondez	$\frac{4 * Área}{\pi * DiametroMax^2}$	1	$\frac{2}{\pi}$
Relación de Aspecto	$\frac{DiametroMax}{DiametroMin}$	1	1
Solidez	$\frac{Área}{Área\ Convexa}$	1	1
Extensión	$\frac{Área}{Área\ Rectangulo\ Envolvente}$	$\frac{\pi}{4}$	1
Compacidad	$\frac{\sqrt{\frac{4 * Área}{\pi}}}{DiametroMax}$	1	$\sqrt{\frac{2}{\pi}}$
Elongación	$\frac{DiametroMin}{DiametroMax}$	1	1

Fuente: SOLOMON, Chris; BRECKON, Toby. Fundamentals of Digital Image Processing. Oxford: John Wiley & Sons, Ltd, 2011. 237p.

4. METODOLOGÍA

Para el desarrollo de este proyecto se hizo una división por etapas las cuales permitirán hacer el diseño e implementación del sistema de monitoreo.

4.1 IDENTIFICACIÓN DE NECESIDADES

En esta etapa se identifican las necesidades del grupo de investigación ambiental GEADES de la Universidad Autónoma de Occidente, que surgen de su estudio sobre el impacto ambiental que sufre la fauna y la flora antes, durante y después de inducir un incendio en un área determinada en la cuenca del río Cali.

4.2 ADQUISICIÓN DE PATRONES PARA RECONOCIMIENTO

Partiendo de la identificación de las necesidades, se procede a recolectar datos pertinentes al proyecto, en este caso sobre las características (morfologicas, color, etc.) de la fauna y flora de un área determinada en la cuenca del río Cali localizada en el jardín botánico. Con los datos recolectados se generan los patrones de reconocimiento utilizados para la identificación de especies de fauna y flora.

4.3 SELECCIÓN DE ALGORITMOS DE PROCESAMIENTO

Con los patrones obtenidos en la etapa anterior, se hace un estudio de técnicas y algoritmos que permitan identificar estos patrones. La escogencia del algoritmo dependerá de su efectividad y su implementación en hardware.

4.4 SELECCIÓN DE PLATAFORMA DE PROCESAMIENTO

En esta etapa se escogerá la plataforma que mejor se adapte a los requerimientos de procesamiento, se hará un cuadro comparativo de las principales características, como capacidad de procesamiento, velocidad, consumo, perifericos, memoria, etc.

4.5 DISEÑO DEL SISTEMA EMBEBIDO

Para el diseño del sistema embebido entran en consideración todos los módulos que componen el sistema de monitoreo y como integrarlos para que su desempeño sea optimo bajo las condiciones externas donde va a operar.

4.6 PRUEBAS Y VALIDACIÓN

Se realizan las pruebas bajo distintas condiciones y escenas, que permitan validar el funcionamiento apropiado del sitema de monitoreo.

5. DISEÑO DEL SISTEMA DE VISIÓN ARTIFICIAL

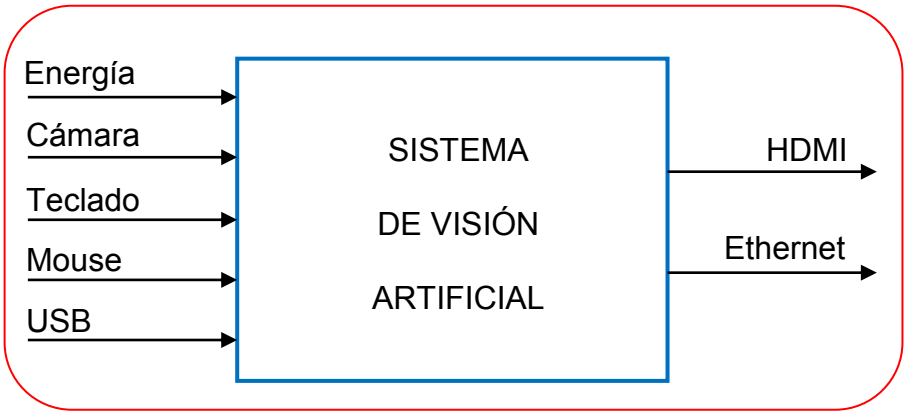
El diseño de sistemas complejos se inicia viendo todo el sistema de forma general y luego descomponerlo en subsistemas que permitan un desarrollo más rápido y eficiente.

En este capítulo se describirá el diseño del sistema de visión artificial partiendo de su forma más general, hasta llegar a la descomposición funcional de cada una de las partes que integra el sistema para escoger la plataforma tecnológica acorde a los requerimientos del diseño.

5.1 CAJA NEGRA

La caja negra es una metodología de diseño que permite visualizar como interactúa el sistema con el medio que lo rodea entendiendo que es lo que hace pero sin dar importancia de cómo lo hace, deben estar muy bien definidas sus entradas y salidas, es decir, su interfaz; en cambio, no se precisa definir ni conocer los detalles internos de su funcionamiento. (Figura 25)

Figura 25. Caja negra



5.2 DESCOMPOSICIÓN FUNCIONAL

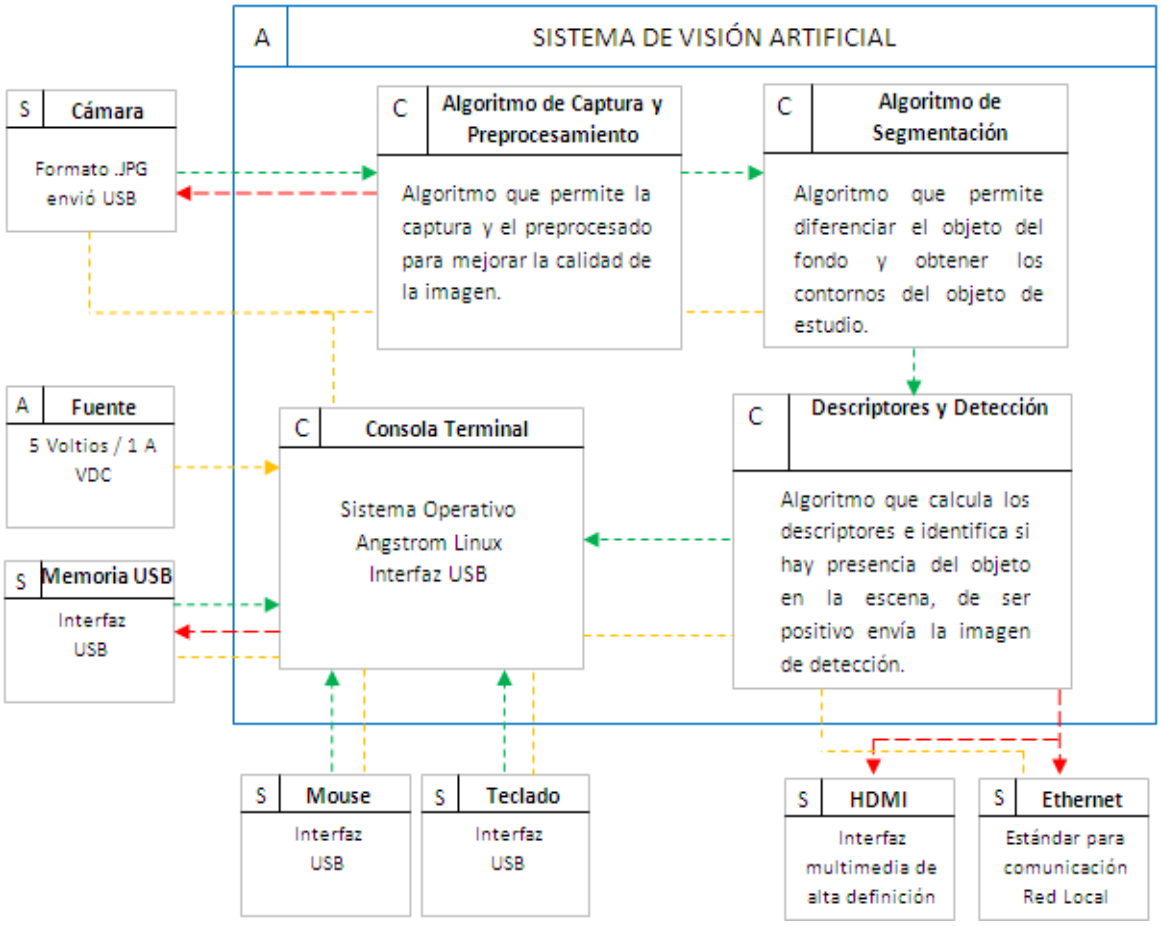
La descomposición funcional es un proceso de la ingeniería concurrente que permite diseñar estructuras modulares y subprocesos para dividir en parte más simples las tareas utilizando criterios y métodos para asignar funciones y establecer conexiones.

Con la ingeniería concurrente orientada al diseño se quiere asegurar que se responden a todas las necesidades, tomar consideraciones desde el inicio de los procesos para la inversión de los recursos y asegurar la calidad del diseño.

En la figura 26 se muestra toda la descomposición funcional del sistema de visión artificial. El sistema se divide en tres procesos, las funciones Activas (A), representan subprocesos que siempre deben estar en continuo funcionamiento para que el sistema opere, Cíclicos (C), subprocesos que se activan por periodos de tiempo y Sensores (S), son los periféricos del sistema, permiten la comunicación con el medio exterior.

Cada subproceso tiene una interfaz que le permite comunicarse entre subsistemas las fechas rojas indican recepción de información, las fechas rojas envió de información y las amarillas distribuye la alimentación por todo el sistema.

Figura 26. Descomposición funcional sistema de visión artificial



5.3 SELECCIÓN DE PLATAFORMA TECNOLÓGICA

Uno de los objetivos del proyecto es lograr implementar un sistema embebido que integre cámara, dispositivo de procesamiento y comunicación, capaz de realizar de forma efectiva la detección y reconocimiento de una araña de la familia *lycosidae*. En otros términos, seleccionar una plataforma tecnológica ideal para la adquisición de imágenes en condiciones determinadas, implementación de algoritmos necesarios para la detección y reconocimiento del individuo y el envío de los resultados a un servidor.

5.3.1 Criterios de Selección. Los criterios utilizados para la escogencia de la plataforma tecnológica fueron, de acuerdo a lo anterior:

Adquisición de imágenes: Una cámara con una resolución mínima para diferenciar la araña de su entorno y compatibilidad con la plataforma de procesamiento tanto en el hardware (conexión) como en el software (controladores y programas).

Dispositivo de procesamiento y comunicación: Un dispositivo capaz de adquirir imágenes de la cámara, ejecutar el programa con los códigos que contienen los algoritmos de procesamiento de imágenes como la transformación y filtrado, la segmentación, el cálculo de características, detección, reconocimiento y decisión. Por último tiene que transmitir los resultados obtenidos mediante un estándar de comunicación compatible con el servidor que albergará los datos, todo esto en él menor tiempo posible ya que se desea acercar lo más posible el monitoreo en tiempo real.

Para la selección de la plataforma tecnológica se dividió en dos partes, conformadas por el dispositivo para el procesamiento y la cámara, a continuación se explicara el proceso llevado a cabo para escoger cada una de las partes mencionadas.

5.3.2 Dispositivos del sistema de visión artificial. Distintos dispositivos de procesamiento de información se utilizaron para realizar pruebas destinadas a determinar cuál era la plataforma más óptima para implementar el sistema de detección y reconocimiento de la araña de la familia *lycosidae.*

Cyclone II - Altera DE2-70 Board

Familia de FPGAs de Altera con procesador NIOS-II de 32 bits utilizado para procesamiento digital de señales, aplicaciones de computación embebidos y sistemas de control.

Entre sus características técnicas sobresalen: 2MB SSRAM, 2 SDRAM 32MB, memoria flash de 8MB, ranura para SD card, oscilador de 50 Mhz, oscilador de 28.63-Mhz para entradas de reloj, controlado de 10/100 Ethernet con conector, conector para cámara de 5mpx y controlador para USB maestro/esclavo.

Figura 27. Cyclone II - Altera DE2-70 Board

Esta fue la primera tarjeta con la que se hicieron pruebas para la implementación del sistema propuesto, su velocidad de procesamiento, la capacidad de trabajar en paralelo y la versatilidad que otorga su programación en software al incluir el procesador, complementado con los módulos hardware modificables, la convirtieron en una opción a considerar.

Al ser una fase inicial del proyecto se decidió primero realizar los códigos de los algoritmos en el programa Matlab para entender y comprobar su funcionamiento de forma más sencilla, logrando mediante códigos como detección de bordes y segmentación por color buenos resultados. Al hacer la migración de lo realizado en Matlab al entorno de la Cyclone II, se hicieron evidentes varios detalles importantes que complicarían el desarrollo del proyecto en esta plataforma.

- **1.** Los algoritmos debían realizarse desde lo más básico, es decir, generar un código propio que contuviera todas las instrucciones necesarias para la detección y reconocimiento del individuo, la razón de esto es que no existen librerías de procesamiento de imágenes optimizadas para el sistema operativo nativo de esta plataforma, de realizarse supondría un incremento significativo de la carga computacional y de la probabilidad de crear conflictos en la ejecución del programa y por ende errores que podrían incluso dañar el dispositivo.

- **2.** La programación de esta FPGA se realiza seleccionando los módulos o bloques que se desean utilizar de la parte física (procesador, memorias, compuertas, cámara, etc.) y posteriormente incluirlas y manipularlas por código, que junto con el punto anterior generan una alta complejidad de programación, como ejemplo está el uso de la cámara para adquirir imágenes, a esta se le debe generar un rutina cíclica en la que tome los datos del entorno o sea las imágenes, transferirlos a una memoria para luego analizarlos y procesarlos, esto hace evidente el arduo trabajo que supone la realización de solo una parte de la implementación.

- **3.** Poca información disponible sobre el desarrollo de aplicaciones para la plataforma en este campo, lo que aumenta la dificultad de programación en ella, cosas como no saber porque aparecen errores en la compilación, no tener casos para comparar o guías, complican el trabajo de desarrollo.

Teniendo en cuenta todos los puntos anteriores y basados en el éxito obtenido con la utilización de la herramienta computacional Matlab la cual tiene una enorme cantidad de librerías y funciones para el procesamiento de imágenes, se tomó la decisión de utilizar una librería de visión computacional compatible con un sistema embebido y con funciones optimizadas básicas para el procesamiento de imágenes , OpenCV fue la librería elegida debido a que es implementable en múltiples sistemas operativos y distribuciones, hay mucha información disponible sobre esta y está orientada aplicaciones de procesamiento de imágenes en tiempo real.

Mini 6410

La tarjeta Mini 6410 surgió como una opción disponible debido a que admite instalar múltiples sistemas operativos embebidos, todos compatibles con la ya mencionada librería OpenCV.

Es una plataforma de desarrollo comercial utilizada para múltiples aplicaciones en sistemas de control con interfaces de usuario.

Sus características más sobresalientes son:

- CPU: 533 MHz Samsung S3C6410A ARM1176JZF-S
- RAM: 128 MB / 256 MB DDR RAM, Bus 32 bit
- Flash: Mas de 1GB NAND Flash
- EEPROM: 256 Byte (I2C)
- Memoria Externa: Ranura SD-Card
- Sistemas Operativos compatibles:
    - Windows CE 6
    - Linux
    - Android
    - Ubuntu
- Ethernet: RJ-45 10/100M (DM9000)

Figura 28. Mini 6410

Antes de realizar cualquier prueba o implementación sobre la Mini 6410 se llevó a cabo la selección del entorno de desarrollo para la realización de los códigos con la librería OpenCV, el entorno que se eligió fue Codeblocks que es un entorno de desarrollo integrado libre y multiplataforma para el desarrollo de programas en lenguaje C y C++ y en el cual existe literatura sobre implementaciones con OpenCV.

Sobre Codeblocks se lograron generar satisfactoriamente los algoritmos previamente hechos en Matlab y de este punto en adelante el resto de los códigos con todo el procesamiento digital de imágenes ya mencionados.

En la Mini 6410 primero se instaló el S.O Windows CE esto debido a que el sistema operativo donde se estaba trabajando era Windows, hallar controladores para Windows es en teoría más probable que para otros sistemas y porque de ser necesario se podría utilizar otro entorno de desarrollo como Visual Studio.

Se procedió a implementar los algoritmos realizados sobre la Mini 6410, pero fue imposible llevarlo a cabo, se generaban errores relacionados con las librerías OpenCV y la ejecución del programa, tampoco se encontró información del significado de los errores o como solucionarlos, al igual que con la Cyclone II no existe información significativa de implementaciones de este tipo sobre esta tarjeta, por lo que se hizo más complicado aún. Múltiples intentos se llevaron a cabo para conseguir algún resultado positivo en la implementación, tales como utilizar el IDE Visual Studio 2008, migrar a diferentes sistemas operativos, entre los cuales se probó Android y Linux con la plataforma de aplicaciones para dispositivos móviles Qtopia, todo esto sin ningún resultado satisfactorio.

Después de indagar e investigar se determinó que los problemas existentes se debían a no poder compilar las librerías de OpenCV en el kernel, lo que derivó en la incapacidad de utilizar la Mini 6410 para la implementación del proyecto.

BeagleBoard- xM

Durante la investigación realizada para los anteriores dispositivos, frecuentemente se encontraron proyectos y aplicaciones para OpenCV en sistemas embebidos. La plataforma más utilizada en estos montajes es la BeagleBoard xM (BB-xM) (figura 29), de Texas Instrument, desarrollada para trabajar con software de código abierto y con múltiples características destacadas como: Procesador TI DM3730 de 1 GHz y núcleo ARM Cortex-A8, Procesador de Gráficos PowerVR SGX 2D/3D, RAM LPDDR de 512MB, Núcleo DSP c64x con capacidad HD, micro SD

de 4GB cargada con la distribución Angstrom (S.O basado en Debian), conectoresDVI-D (HDMI), 4 puertos USB, puerto para cámara, puerto Ethernet, ranura para tarjeta micro SD/MMC, entre otras.

Figura 29. BeagleBoard-xM

En comparación con la Cyclone II y la mini 6410, acerca de la BB-xM se puede hallar mucha más información en cuanto a su programación y desarrollo en programas de visión artificial basadas en la librería OpenCV, y lo más importante, permite de manera sencilla, incluir en el kernel del sistema operativo los compiladores y bibliotecas específicos con los que se desea trabajar, que fue el principal inconveniente cuando se trabajó con la Mini 6410.

Consiguiendo la inclusión de las librerías de OpenCV y los compiladores para estos, se logra hacer una migración e implementación exitosa de los algoritmos de visión artificial planteados previamente en el PC, sobre una plataforma embebida que realiza de manera automática los procesos de adquisición de imágenes, procesamiento de estas para su identificación y a la vez hace el papel de servidor para el acceso a las imágenes de detecciones positivas vía red.

Cámara

La selección de la cámara se hizo teniendo en cuenta una resolución mínima para diferenciar la araña de su entorno y compatibilidad con la plataforma de procesamiento.

Se evaluaron cámaras web de diferentes resoluciones y calidades, buscando obtener una imagen lo más clara y nítida posible. Se debe tener presente que una buena imagen no es resultado solo de una buena resolución del sensor de imagen, también está relacionado con el tipo de sensor utilizado (CMOS o CCD) y la calidad en su fabricación.

El dispositivo escogido fue una cámara web Microsoft HD-300, su resolución puede ser modificada desde 160x120 píxeles hasta 1280x720 píxeles, presentó la mejor respuesta a las variaciones de iluminación, su base modificable permitió acomodarla al montaje de pruebas y es compatible con la BeagleBoard-xM.

Características:

- USB 2.0
- Base modificable
- Resolución: 1280x720 píxeles
- Aplicaciones web en HD

Figura 300. Cámara Microsoft HD-300

Enrutador Inalámbrico

La comunicación de los resultados es uno de los puntos importantes en los objetivos del proyecto, ya que al haber una detección positiva de uno de los individuos del estudio, se deber tomar la imagen y montarla en un servicio de red, al cual los investigadores del grupo GEADES deben ser capaces de acceder.

La selección del dispositivo a utilizar se llevó a cabo principalmente de acuerdo a la compatibilidad con el sistema operativo de la BB-xM y la disponibilidad de la tecnología. Se plantearon varias alternativas como un Access Point o Modem USB, pero la mayoría requieren un driver y un software para funcionar, estos por lo general están disponibles solo para los sistemas operativos más comunes. Tampoco se tuvieron disponibles este tipo de dispositivos para realizar pruebas a profundidad. La siguiente alternativa sugerida fue la utilización de un enrutador inalámbrico para enviar los datos o imágenes a través de WIFI. Estos equipos traen incluido su propio software para la gestión de red al cual se accede desde los dispositivos miembros de la red. La conexión con la BB x-M se hace por medio de Ethernet, estándar compatible con la plataforma, esto evita la necesidad de instalar un driver específico o un software.

Se seleccionó un Router inalámbrico D-Link DI-524 (Figura 31), principalmente por la disponibilidad que se tenía para poder utilizarlo. Sus características son:

- Opera bajo el estándar Wi-Fi 802.11g
- Hasta 54 Mbps de velocidad en la transmisión de datos.
- Firewall avanzado y control parental.
- Estándar compatible con 802.11b.

Figura 31. Router Inalámbrico D-Link DI-524

5.4 IMPLEMENTACIÓN Y COMUNICACIÓN

En este capítulo se presentarán todos los métodos que permitieron la implementación y comunicación entre los subsistemas para el desarrollo del sistema de visión de artificial, basado en la descomposición funcional ver figura 26.

Para el desarrollo de los algoritmos de procesamiento se planteó la necesidad del uso de una librería y un IDE que permitiera la ejecución y depuración del código "offline", lo que significa desarrollar los algoritmos en un PC sin necesidad de correrlos en la plataforma. Una vez los algoritmos son depurados, estos deben implementarse sobre el embebido, por eso la necesidad de una librería que fuera multiplataforma.

5.4.1 OpenCV y Code Blocks. OpenCV es una librería bajo licencia BSD y por lo tanto es libre para uso académico y comercial. Cuenta con C, C++, Python y las interfaces de Java, es compatible con Windows, Linux, Mac OS, iOS y Android. OpenCV fue diseñado para la eficiencia computacional y con un fuerte enfoque en las aplicaciones en tiempo real. El entorno de desarrollo es fácil de utilizar y altamente eficiente, su programación está basada en código C y C++ optimizados.

Code Blocks es un entorno de desarrollo gratuito, optimizado para C++ ya que brinda facilidades y comodidades a la hora de trabajar con este tipo de lenguaje. Está basado en la plataforma de interfaces gráficas WxWidgets, lo cual quiere decir que puede usarse libremente en diversos sistemas operativos, y está licenciado bajo la Licencia GNU.

La principal razón del uso de este entorno de desarrollo fue la facilidad que presenta al integrar las librerías de OpenCV frente a otros IDEs como Visual Studio y Dev-C++. Con estas dos herramientas enlazadas se realizó el estudio e implementación de los algoritmos de detección de la araña y los brotes de flora bajo un sistema operativo Windows. Ver Anexo A *"Guía de Instalación de OpenCV y Code Blocks"*.

Una vez desarrollado los algoritmos estos se deben migrar hacia la Beagle Board, para esto se deben seguir una serie de procedimientos que permitan integrar las librerías de OpenCV con el Kernel del sistema operativo.

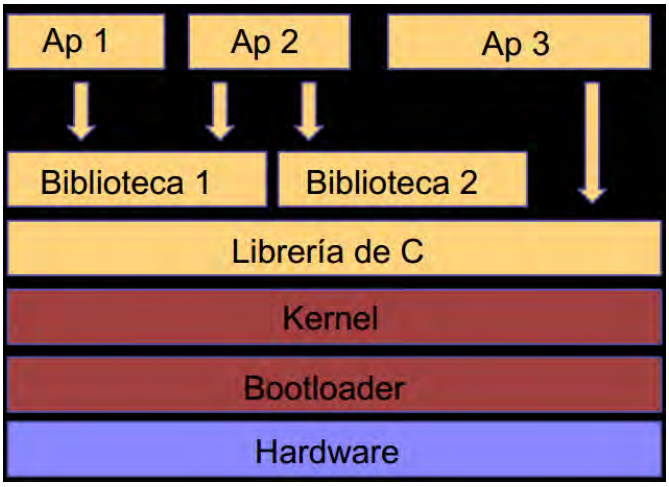
5.4.2 Angstrom. Linux embebido o empotrado se refiere al uso del sistema operativo Linux en un sistema embebido, como por ejemplo PDA, teléfonos móviles, robots, enrutadores /servidores, dispositivos electrónicos y aplicaciones industriales con microcontroladores y microprocesadores.

Angstrom es una distribución de Linux para sistemas embebidos, la principal razón del uso de este SO en plataformas, es la de reducir la complejidad del desarrollo del producto y lograr adaptar el Kernel a las necesidades del diseño.

Cuando se trabaja con sistemas embebidos, el desarrollo de las aplicaciones tiene limitaciones debido a que el sistema posee recursos limitados de memoria y velocidad de procesamiento, la ventaja de usar estos SO es que permite trabajar con lenguajes de alto nivel para el desarrollo de las aplicaciones como Python y C++. Otra de las ventajas de usar Linux es la gran variedad de lenguajes y toolkits gráficos que se pueden emplear.

En la figura 32, se muestra un diagrama de bloques que permite visualizar como se desarrollan las aplicaciones y como se integran con el SO. Las aplicaciones pueden trabajarse directamente en lenguaje C o llamar bibliotecas como OpenCV para el desarrollo de las aplicaciones.

Figura 32. Diagrama de bloques para integrar aplicaciones en Linux



Una de las tareas más complejas es poder integrar el sistema con las librerías para el desarrollo de las aplicaciones, para esto se debe seguir una serie de procesos como: seleccionar los compiladores adecuados, manejar la instalación

de todos los paquetes de software, manejar las dependencias entre paquetes, aplicar los parches necesarios y configurar las fuentes.

Como parte del desarrollo de la tesis, se diseñó una guía explica de manera detalla la forma de integrar Angstrom con las librerías de OpenCV, como se deben configurar las fuentes y el compilador para generar las aplicaciones, ver Anexo B *"Guía para integrar OpenCV con el SO Angstrom".*

5.4.3 Servidor Web. Uno de los objetivos del proyecto es poder transmitir las imágenes cuando se consiga una detección exitosa de la araña en la escena, para esto se utilizaron las ventajas que brinda el sistema embebido.

La Beagle-Board cuenta con un puerto de Ethernet emulado sobre USB el cual se puede modificar para poder trabajar como un servidor en una Red LAN. Para prestar el servicio de HTTP, se utilizó el servidor HTTP de Apache, este es usado principalmente para enviar páginas web estáticas y dinámicas en la World Wide Web. Sus principales características son:

- Código abierto para plataformas UNIX, Windows y Macintosh.

- Es altamente configurable

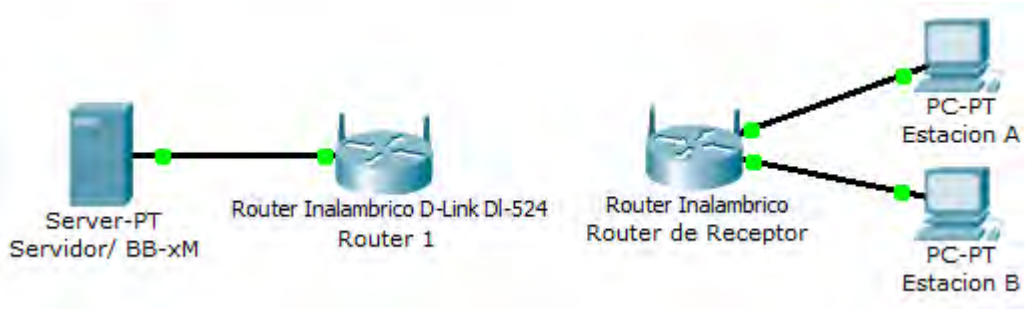
- Bases de datos de autenticación y negociado de contenido.

En la figura 33 se muestra el esquema de la red LAN propuesta, el servicio HTTP lo provee la Beagle Board y la comunicación se realiza punto a punto con el router inalámbrico, para esto se emplea un cable cruzado de Ethernet RJ-45.

La conexión a la red por medio de los usuarios se realiza de forma inalámbrica (WIFI). Los dispositivos se pueden conectar a la página principal a través de un punto de acceso de red inalámbrica. Dicho punto de acceso (o hotspot) tiene un alcance de unos 20 metros en interiores y al aire libre una distancia mayor.

En el Anexo C, *"Guia de configuración servicio HTTP Apache",* se presenta una guía que permite configurar las direcciones IP de la Beagle Board y la configuración del servicio de HTTP en el servidor Apache.

Figura 33. Esquema de una Red LAN Inalámbrica



6. RESULTADOS EXPERIMENTALES

Este capítulo muestra los resultados obtenidos durante el desarrollo del diseño del sistema de detección e identificación de una especie de araña en un ambiente alterado por un incendio y la detección de posibles brotes de flora, basado en un sistema de visión computacional implementado en un dispositivo de procesamiento embebido.

Los resultados obtenidos se dividen en dos módulos según lo siguiente:

- **1. Pre-Implementación**: Este módulo corresponde a toda la investigación y experimentación llevada a cabo antes de implementar el programa y rutinas de detección sobre el sistema embebido. Corresponde entonces a trabajo hecho sobre un PC.

- **2. Implementación en plataforma:** En este módulo se realizaron todas las pruebas y ajustes concernientes a la implementación del programa de detección sobre la plataforma hardware (BeagleBoard-xM).

6.1 PRE-IMPLEMENTACIÓN

En este módulo se presentan los resultados obtenidos en las etapas relacionadas a la construcción del programa de identificación planteado y la comprobación de su funcionamiento en un PC.

6.1.1 Detección de Brotes de Flora. El escenario en el que se va a llevar a cabo la detección consiste en un terreno quemado a nivel del suelo, este tiene por característica una tonalidad oscura o grisácea. Se puede deducir entonces que una propiedad sobresaliente para un brote de vegetación en una escena de este tipo serían sus colores verdes claros vivos.

Se escogió como descriptor para los brotes de vegetación el color, buscando una mínima cantidad de píxeles con un valor de color establecido en un rango, agrupados en una región. Se decidió utilizar esta característica como herramienta de segmentación y a la vez como descriptor único para los brotes, debido a que existen una gran cantidad de tipos de plantas y de variaciones en su germinación, haciendo muy difícil una caracterización orientada a patrones geométricos.

Para la segmentación se buscaron modelos de color disponibles en la librería de visión computacional de Opencv y que además tuvieran independencia entre sus componentes de intensidad de luz y la cromaticidad, con el propósito de disminuir la incidencia de la luz en los valores de color. Las opciones que se consideraron fueron el modelo HSV y el YCbCr.

Se hicieron una serie de pruebas experimentales para ver cuál de estos dos modelos representaba mejor los datos de color del brote, a continuación se exponen los resultados de una de las múltiples pruebas realizadas para una imagen de un brote.

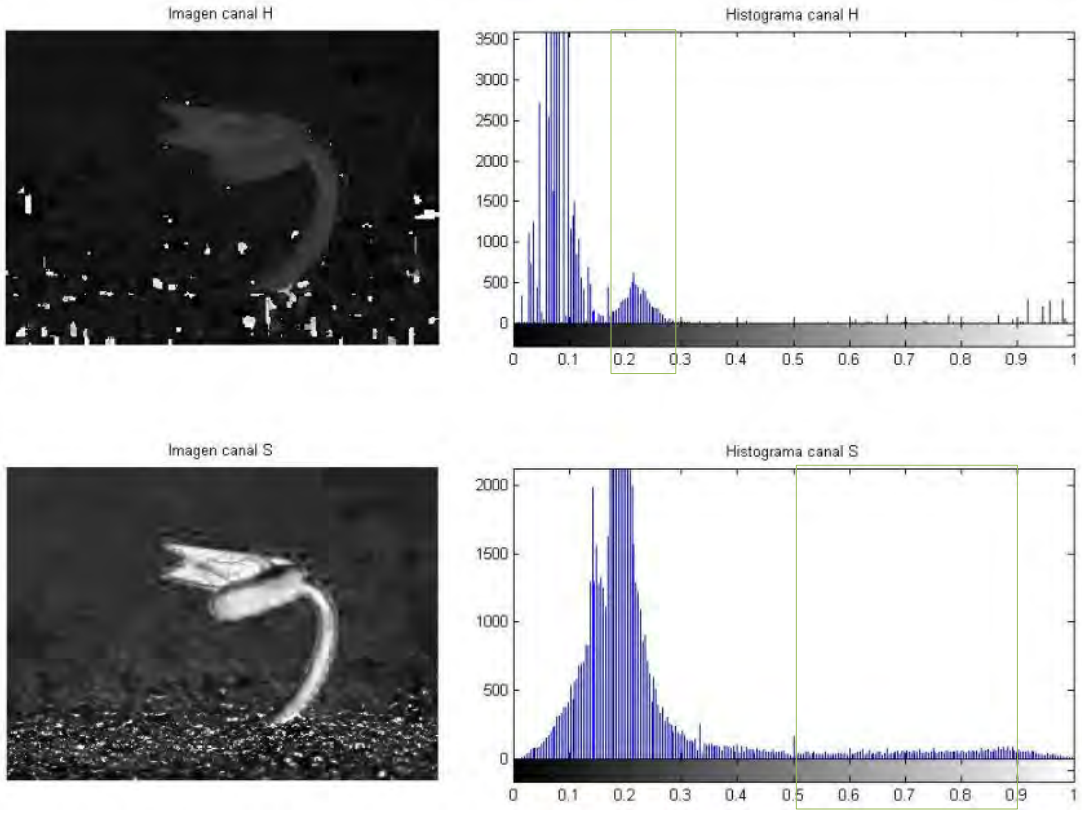
YCbCr Vs. HSV

Las pruebas consistieron en hacer una visualización de la imagen de la figura 34 en los diferentes canales con componentes de color de los modelos YCbCr y HSV, todo esto para saber si se podría tener una apreciación clara del brote de vegetación. Junto con esto se graficaron los histogramas de cada canal, para determinar si era posible detectar al sujeto en algún rango de valores de color, pensando en hacer una segmentación basados en esos valores.

Figura 34. Imagen de Brote de Prueba

En la figura 35 se exponen a la izquierda los planos de imagen de color del modelos HSV, correspondientes al tono (H) y saturación (S), a la derecha sus correspondientes histogramas.

Figura 35. Canales H y S con sus Histogramas en Imagen de Brote



En el análisis para la componente H se puede observar que el brote tiene una tonalidad media entre el fondo, que es oscuro, y los valores más cercanos al blanco. Si se hace una segmentación por umbral de acuerdo al histograma, el brote correspondería a los valores encerrados en el recuadro del histograma del canal H de la figura 35.

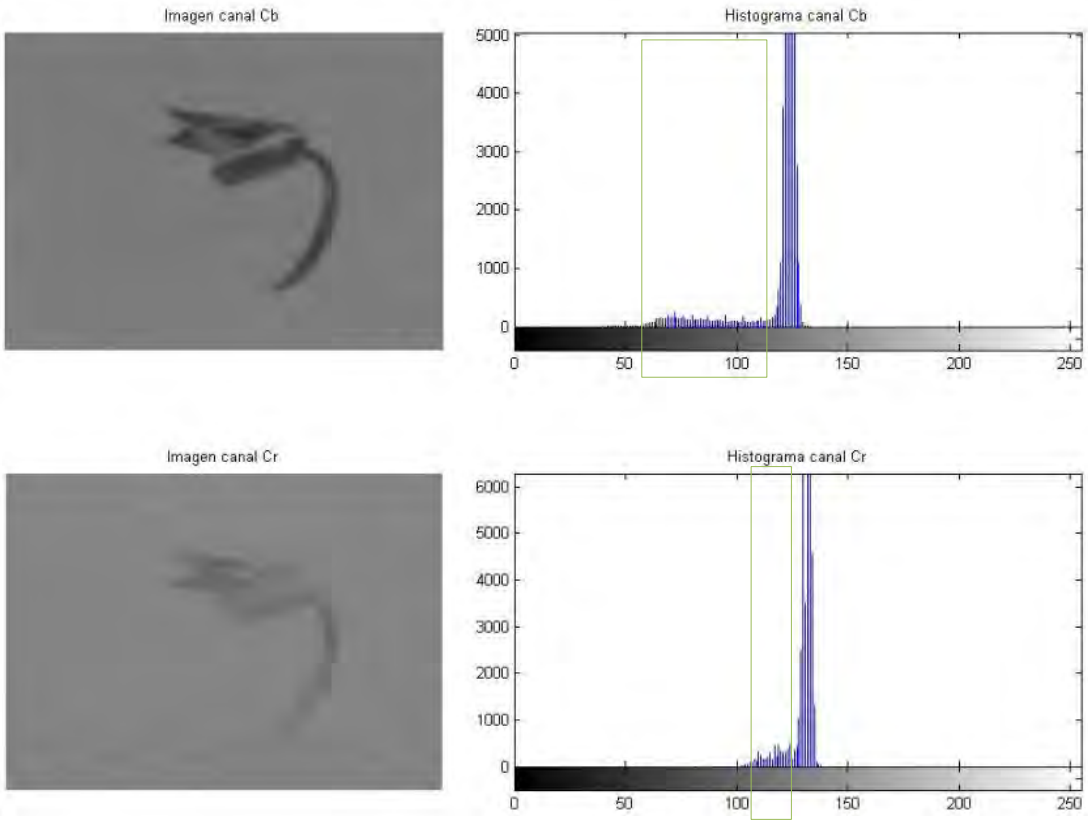
Por parte del canal S, se puede ver que el brote es más claro, por lo que sus valores están más cercanos al blanco, si se hace una segmentación, el brote correspondería a los valores encerrados en el recuadro mostrado en el histograma del cana S de la figura 35.

De acuerdo a lo anterior si se lleva a cabo una segmentación por umbral entre los valores que se muestran en cada histograma, el brote debe aparecer en blanco y el resto en negro. En la figura 36 se muestra a la derecha la segmentación para el canal H y a la izquierda la correspondiente al canal S.

Figura 36. Segmentación canales H y S para imagen de Brote

Para el modelo YCbCr las componentes de color son los canales Cb y Cr, que hacen referencia a la cromaticidad presente en la imagen, en la figura 37 se exponen las imágenes representativas de los canales Cb y Cr, con sus respectivos histogramas de imagen.

Figura 37. Canales Cb y Cr con sus Histogramas en Imagen de Brote



La imagen del canal Cb que se muestra en la figura 37 permite ver al brote diferenciado del fondo, por sus tonalidades más oscuras. Haciendo un análisis como el realizado para los canales H y S, teniendo como objetivo una segmentación por umbral, se estudian los datos otorgados por el histograma correspondiente al canal Cb, rápidamente de pueden identificar los valores pertenecientes al fondo, porque es casi uniforme y porque hay más píxeles de este que del brote, por lo que la germinación de la planta serían los valores que se encuentran en el recuadro del histograma correspondiente al canal Cb de la figura 37.

En cuanto al canal Cr, se consigue una imagen similar a lo obtenido en el canal Cb, pero la silueta del brote es más tenue, una explicación para este fenómeno es que al ser el canal Cb la componente de azul, este se encuentra más cerca al verde en el espectro de colores que del rojo (Cr). Para una segmentación, se puede identificar en el histograma los valores probables del fondo y del brote siguiendo el mismo razonamiento que para el canal Cb. En el recuadro que está en el histograma perteneciente al canal Cr se encuentran los valore que corresponden al brote.

Al realizar la segmentación del brote según los valores descubiertos en los histogramas, el resultado obtenido es el mostrado en la figura 38, a la izquierda la segmentación del canal Cb y a la derecha el correspondiente al canal Cr.

Figura 38. Segmentación canales Cb y Cr para imagen de Brote

Selección de Modelo de Color

Una vez obtenidos los resultados de las segmentaciones para las componentes de color de los modelos HSV y YCbCr para esta prueba sencilla, y basados en otras

pruebas similares hechas con otras imágenes, se seleccionó al modelo YCbCr y sus componentes de color para utilizarlos en la detección de brotes de vegetación.

Las principales razones fueron:

- Las segmentaciones realizadas para los componentes del modelo HSV incluyeron constantemente valores no deseados en la segmentación para las pruebas realizadas, en la figura 36 se puede apreciar un ejemplo de dichos valores. Esto puede ocasionar falsos positivos al momento de hacer una detección.

  Los resultados obtenidos con el modelo YCbCr fueron mucho más sólidos en relación a la inclusión de ruido o valore indeseados, en la figura 38 se puede ver una segmentación consistente, acorde con el objetivo que se busca.

- El análisis de los histogramas permitió ver en el caso del canal H que si bien es posible determinar los rangos de valores para el brote, estos suelen quedar en zonas intermedias por la constante inclusión de valores indeterminados en este canal lo que dificulta relativamente la detección.

  El canal S muestra una distribución muy grande de los valores correspondiente al brote, incluyendo valores de colores de otros objetos y del terreno, esto provoca que aparezcan puntos aislados que podrían considerarse como ruido.

  Para los canales Cb y Cr, la distribución de valores es mucho más concentrada, esto se traduce en que es más probable que de aparecer objetos con otros colores en la escena, estos tengan valores diferentes a los del brote.

Imágenes de Comprobación

La figura 39 expone algunas imágenes de brotes en diferentes escenarios, cada una pasó por el proceso de segmentación por umbral con los mismos valores para todas, el resultado es el que aparece al lado de cada imagen. La segmentación se hizo para el canal Cb.

Figura 39. Imágenes de Brotes con Segmentación para Canal Cb

6.1.2 Detección de Araña. El programa o código desarrollado para la detección de la araña, se basó principalmente en los conceptos básicos de los sistemas de visión artificial como los vistos en el capítulo 3, apartado de algoritmos de procesamiento. Todos esos algoritmos son el resultado lógico que se obtiene al intentar que una maquina interprete lo que hay en una imagen determinada y pueda cumplir con un objetivo como el de identificar un objeto.

El diagrama de bloques de un sistema de visión artificial (figura 8) ilustra los componentes secuenciales que se utilizan en sistemas de estas características para la detección de un objeto, estos mismos fueron aplicados en este proyecto para poder identificar a una araña en un zona de pos fuego.

Para validar los procesos realizados en la construcción del programa de detección, a continuación se muestran los resultados más significativos que se obtuvieron en este módulo de pre-implementación del sistema para la detección de la araña.

Selección de Descriptores

El primer paso realizado para la conformación del programa de identificación de la araña, fue encontrar los descriptores cuyos valores la distinguieran de otros objetos presentes en la escena de detección. Además se tuvieron en cuenta varios criterios acorde con la propuesta de un sistema robusto e implementable en un dispositivo embebido. Los criterios son:

- Insensibilidad a variaciones como cambios de tamaño, traslación y rotación.
- Poca memoria para almacenar los descriptores.
- Poco coste en la extracción del descriptor y en el cálculo de similitud.
- Poca complejidad algorítmica para poder ser implementado.

Con lo anterior presente se procedió a encontrar los descriptores para la identificación de las arañas.

Las pruebas tempranas consistieron en crear una base de datos con imágenes de la araña sobre un fondo blanco (ejemplo figura 40) y condiciones de iluminación establecida, se hizo bajo estas condiciones porque el objetivo era extraer los valores de los descriptores de la araña, sin necesidad realizar un proceso de segmentación innecesariamente complicado. Igualmente se realizó una base de datos con objetos que probablemente pudiesen aparecer en la zona de detección

del sistema (ejemplo figura 41), bajo las mismas condiciones mencionadas para saber al compararlos cuales eran los más idóneos.

Figura 40. Imagen Ejemplo: Araña en Fondo Blanco

Figura 41. Imagen Ejemplo: Objetos en Fondo Blanco

Como se mencionó en el tema de descriptores, existen dos géneros que los abarcan, los de representación interna y los de representación externa. Basados en los criterios se decidió evaluar primero los descriptores internos (área, momentos, esqueletos, etc.), ya que estos son mucho más fáciles de implementar y son menos complejos computacionalmente que los descriptores externos (códigos de cadena, descriptores de Fourier, aproximaciones poligonales, etc.).

Los descriptores evaluados tanto para la araña como para los objetos fueron: Factor de Forma, Redondez, Relación de Aspecto, Solidez, Extensión, Compacidad, Elongación y Momentos Invariantes.

Para obtener los valores de los descriptores se generó un código en el programa Matlab diseñado para extraer automáticamente los valores de las características de las imágenes que se le indiquen y colocarlos en una tabla. Los datos tabulados se graficaron, para ver la distribución de los valores de los descriptores de la araña y de los objetos.

Las gráficas obtenidas para los descriptores de arañas y objetos sobre fondo blanco se muestran a continuación. Los datos en rojo corresponden a la araña, en verde los valores para objetos aleatorios como rocas, hojas, ramas, entre otros.

Figura 42. Gráfica Descriptor: Factor de Forma

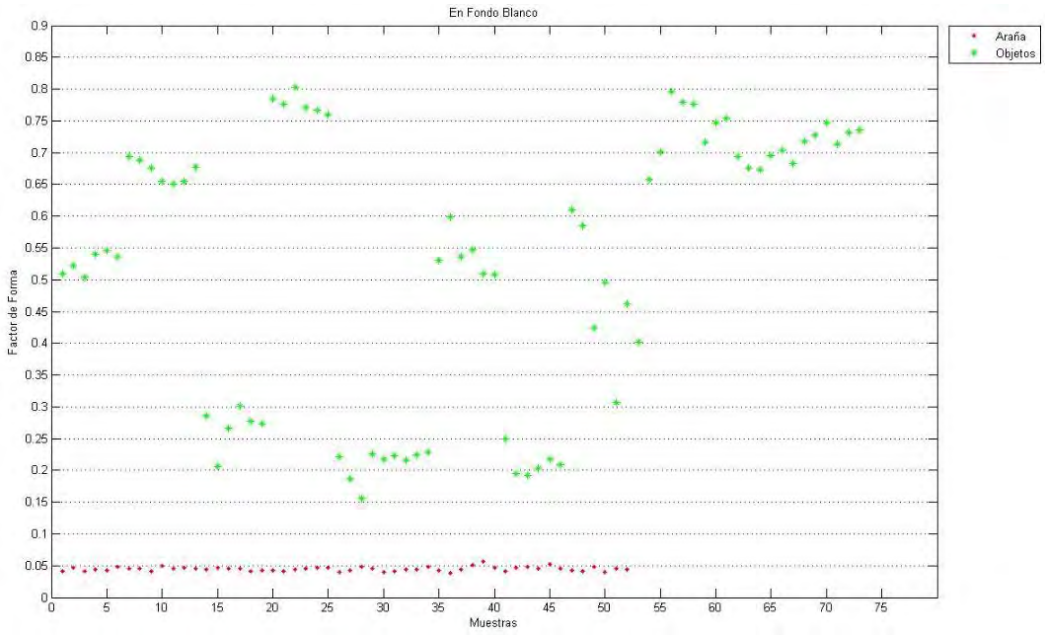


Figura 43. Gráfica Descriptor: Redondez

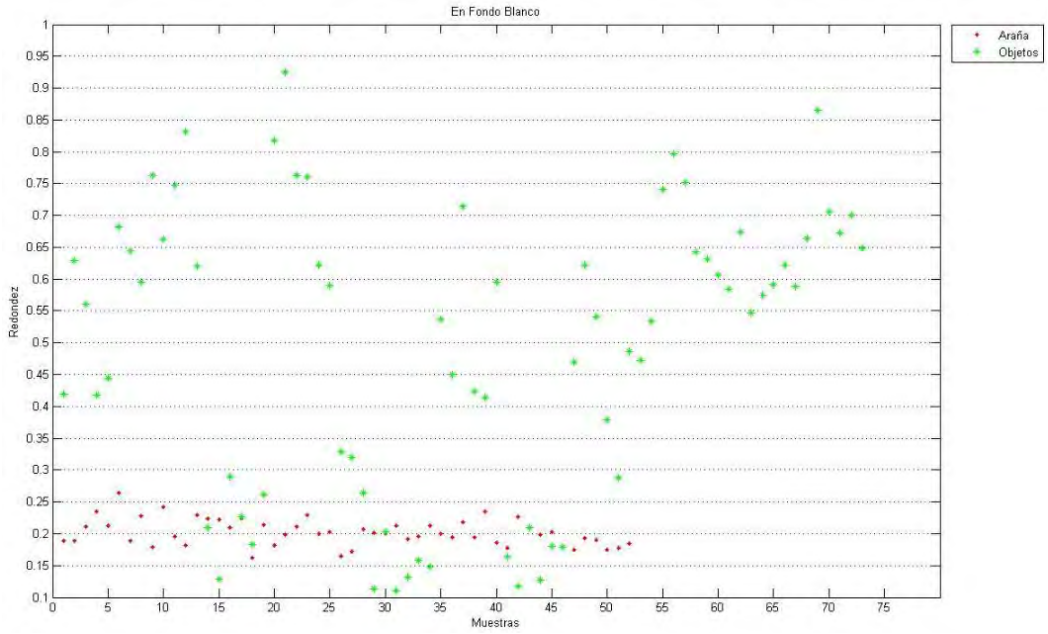


Figura 44. Gráfica Descriptor: Relación de Aspecto

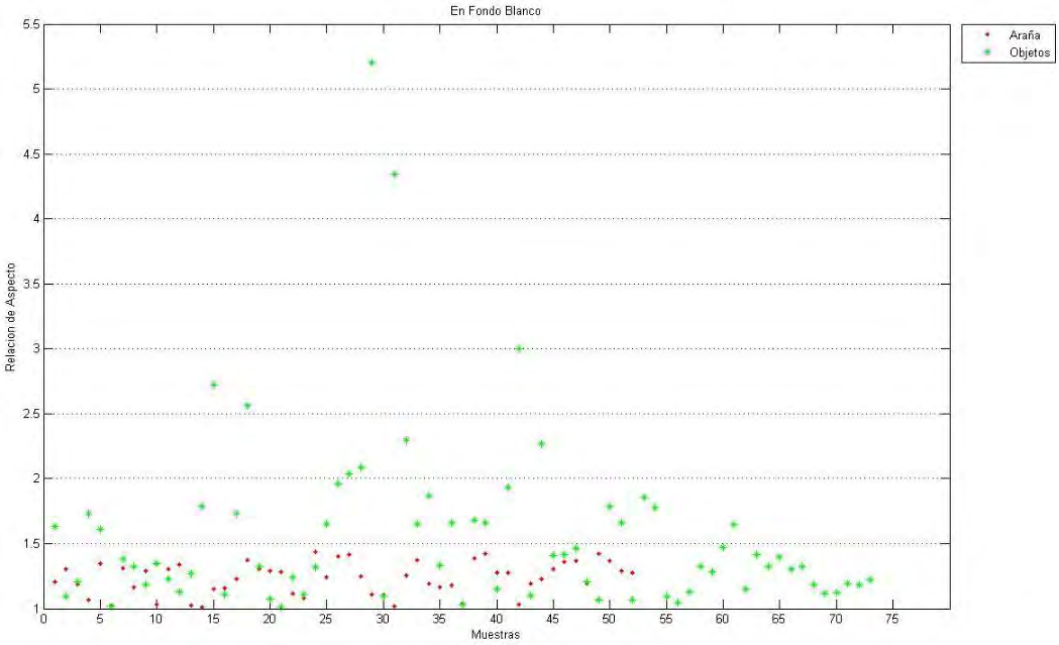


Figura 45. Gráfica Descriptor: Solidez

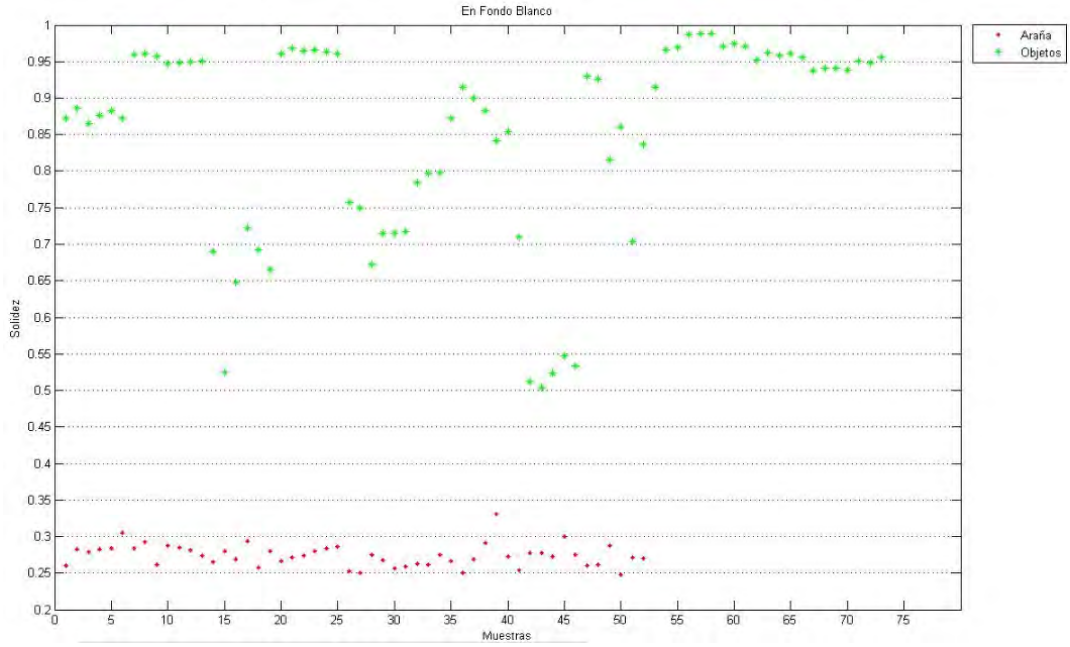


Figura 46. Gráfica Descriptor: Extensión

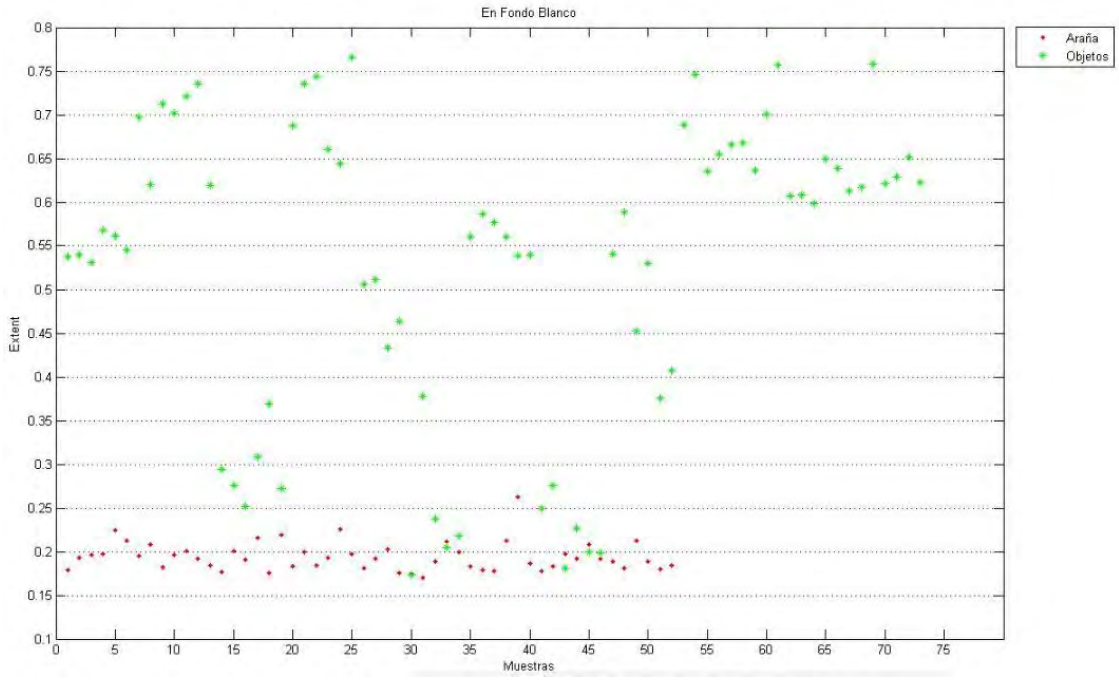


Figura 47. Gráfica Descriptor: Compacidad

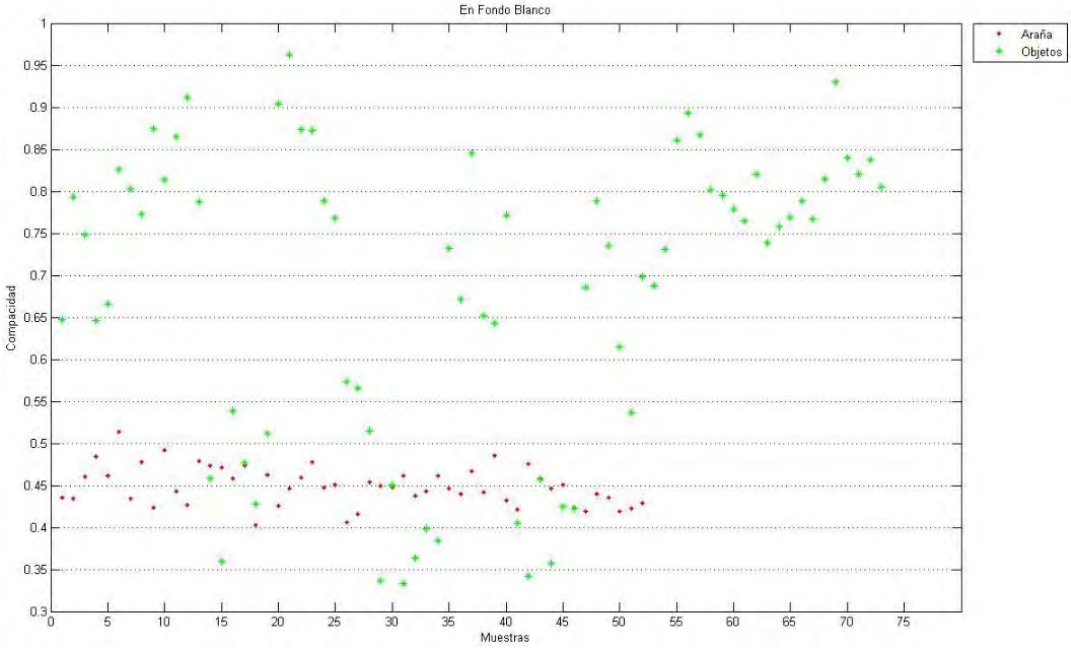


Figura 48. Gráfica Descriptor: Elongación

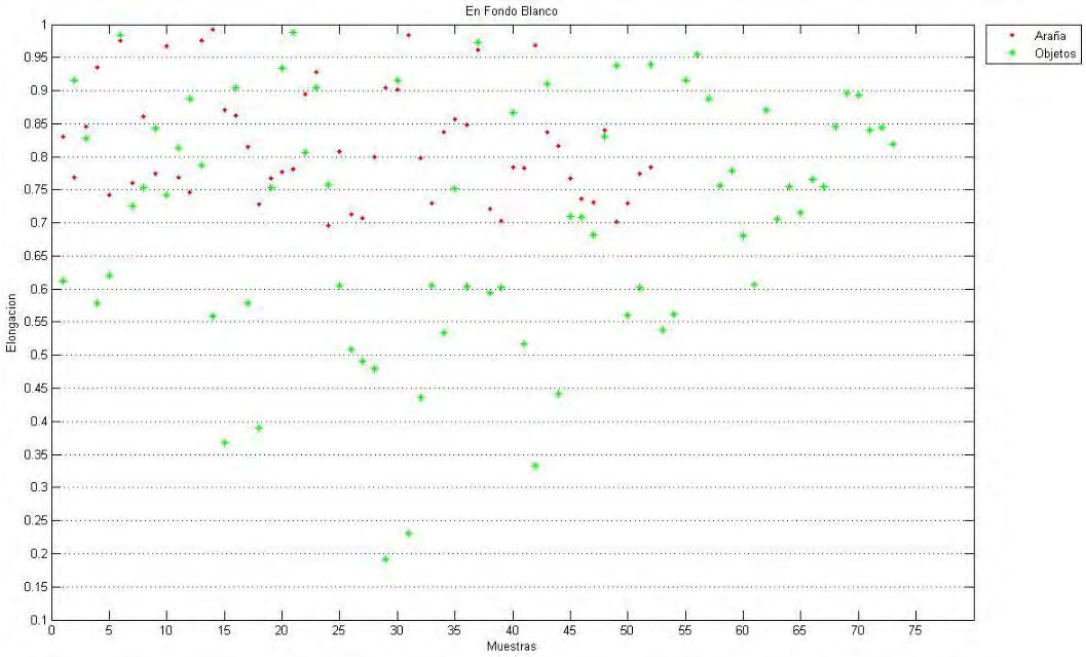
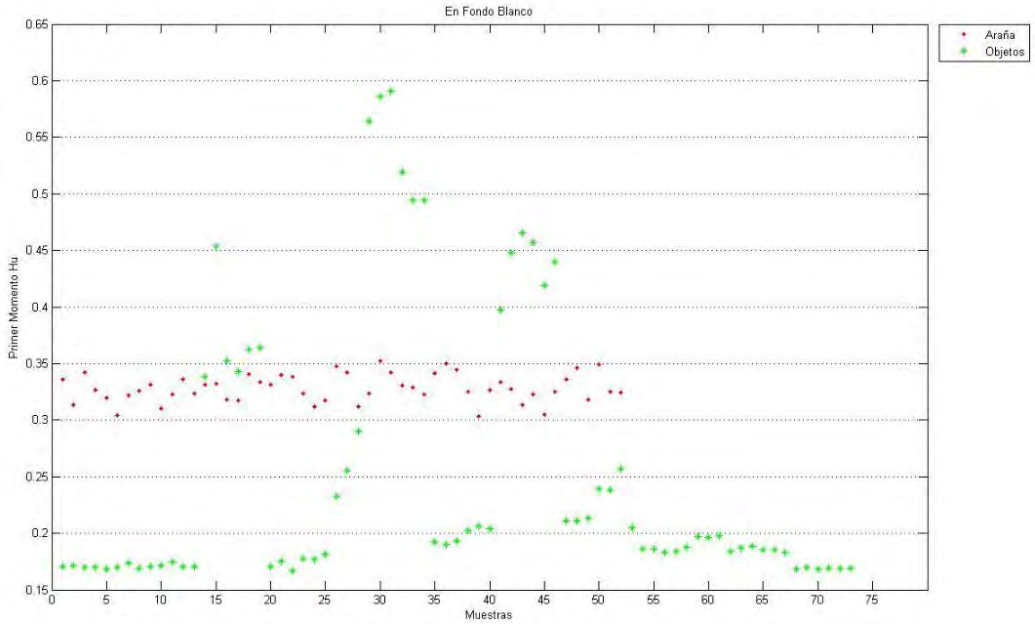


Figura 49. Gráfica Descriptor: Primer Momento Hu



En las gráficas obtenidas se puede apreciar que hay tres descriptores que por probabilidad serían los más adecuados para describir a la araña por sus características de forma en comparación con otros objetos, estos descriptores fueron: Factor de Forma (Figura 42), Solidez (Figura 45) y Momentos Invariantes (Figura 49).

Los tres descriptores internos mencionados fueron seleccionados para ser los patrones de detección, la cual es por coincidencia de valores en los resultados con un rango de tolerancia.

Métodos de Segmentación

Para llevar a cabo la segmentación de los objetos presentes en la escena fotografiada se evaluaron dos casos: Fondo Claro y Fondo Oscuro. Se hizo de esta forma para estudiar los descriptores y las técnicas de segmentación, primero experimentando con ellas en un fondo claro, en donde discernir la araña del fondo era más sencillo y después con una escena más compleja donde el fondo oscuro se asemeja al escenario de detección con terreno quemado.

Es necesario indicar que este tipo de arañas tienen características nocturnas, por lo que las pruebas realizadas fueron llevadas a cabo bajo condiciones de oscuridad ambiental e iluminación artificial de apoyo, compuestas por dos leds con capacidad de hasta 5W, regulados dependiendo de la aplicación.

Segmentación en Fondo Blanco

La técnica empleada para la segmentación en fondo blanco está basada en el modelo de sustracción de fondo, el objetivo es detectar los pixeles que no pertenecen al fondo de la imagen.

El primer paso es capturar una imagen del fondo la cual es guardada para realizar una operación de división con una nueva captura, el resultado es una imagen que muestra los pixeles que presentaron mayor cambio de posición o movimiento en la escena. Ver figura 52.

El siguiente paso es realizar una umbralización que permita diferenciar el objeto del fondo, para determinar el valor óptimo de umbral se realizaron diferentes pruebas con imágenes de arañas en fondo blanco. Ver figura 53.

La última etapa es realizar el cierre de los contornos, este proceso morfológico permite cerrar los pequeños espacios que se generan al realizar la umbralización, además de eliminar los pequeños objetos que no hacen parte del contorno de la araña. Ver figura 54.

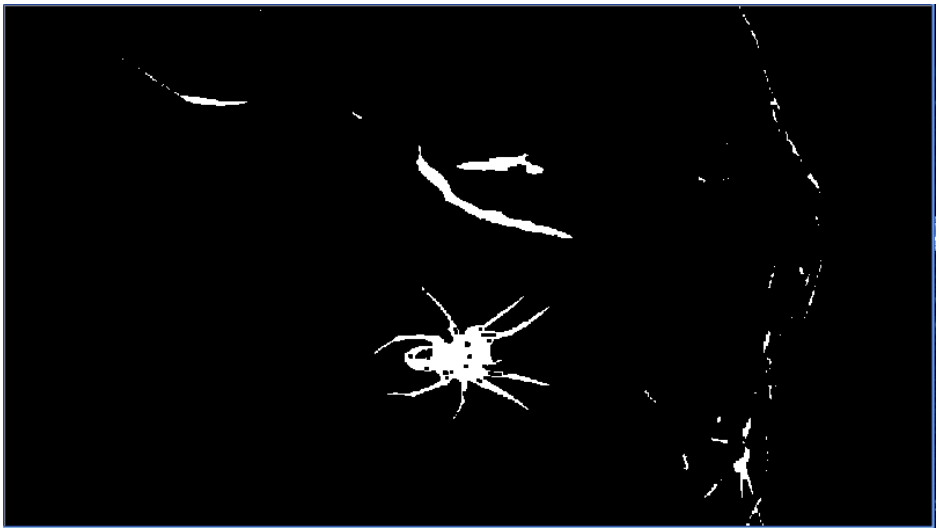
Figura 50. Imagen de Captura del Background

Figura 51. Imagen de Captura de Foreground

Figura 52. Operación de división entre el Background y Foreground

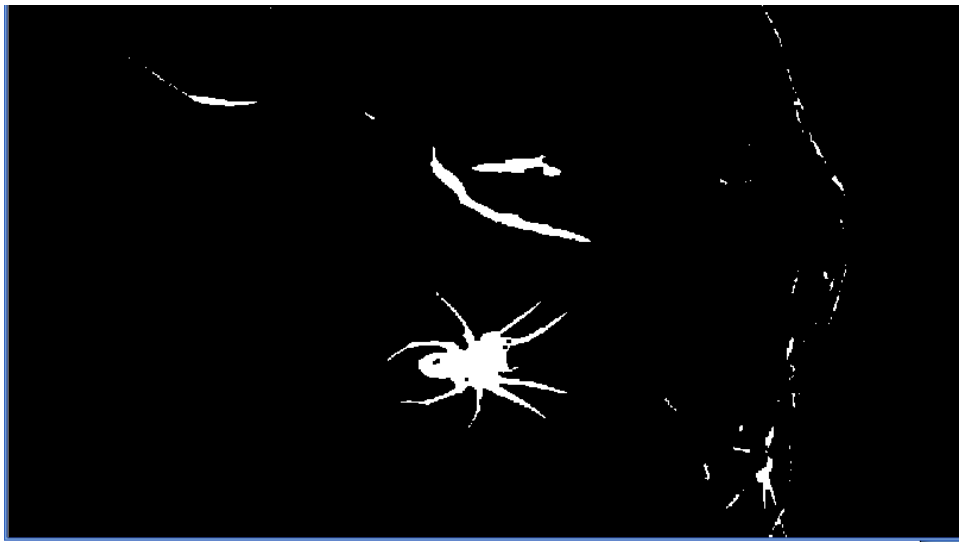
Caption: Se realizó operación de división entre el Background y Foreground la función que permite la operación en OpenCV es cvDiv(Img_In ,Img_Out ,Img_Resultado).

Figura 53. Umbralización del objeto



Caption: Valor de umbral = 190, función de OpenCV para realizar la umbralización es cvThreshold(Img_In, Img_Out, Umbral, 255, Threshold_type), donde Threshold_type, puede ser Binary, Binary-Inv, Otsu.

Figura 54. Operación de Cierre de Contornos



Caption: Se realizó operación morfológica cierre de Contornos, la función de OpenCV para realizar la operación es cvMorphologyEx(img_In, img_Out, 0, NULL, CV_MOP_CLOSE, 1), CV_MOP puede ser ERODE, DILATE y OPEN.

Segmentación en Fondo Negro

Al intentar aplicar las técnicas utilizadas para segmentar en fondo blanco sobre las imágenes en fondo negro (Ejemplo Figura 55), los resultados de los experimentos no resultaron satisfactorios, aunque se variaron los parámetros de la umbralización y se probaron técnicas más avanzadas como umbralización adaptativa con condiciones de iluminación controlada, los valores obtenidos para la araña se acercaban mucho en algunas regiones a los del fondo, por lo que la segmentación de la araña resultaba con mucho ruido o se eliminaban partes de esta, haciendo que los datos de los descriptores no coincidieran con los de la caracterización (Figura 56).

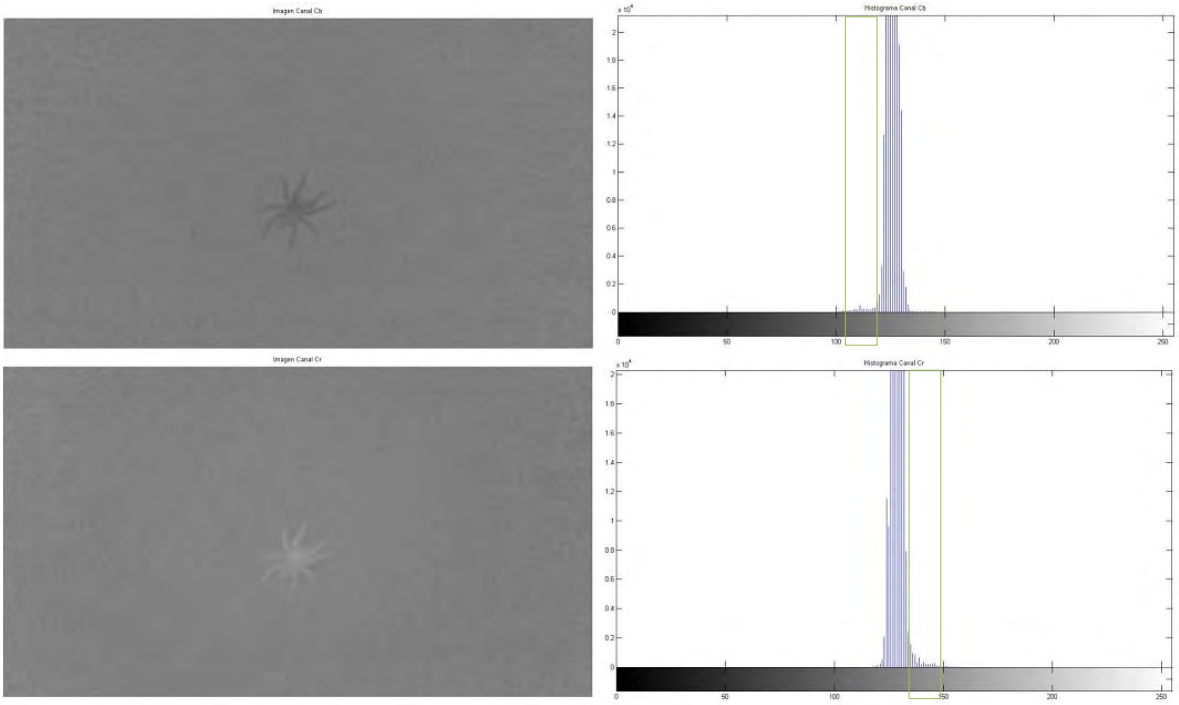
Figura 55. Ejemplo Araña Sobre Fondo Negro

Figura 56. Ejemplo Segmentación para Fondo Blanco en Fondo Negro

Se hizo necesario buscar una forma distinta de segmentar la araña del fondo negro que permitiera extraer el contorno de la araña y hacer viable la valoración de los descriptores.

Utilizando la experiencia obtenida en la segmentación de los brotes de vegetación, se decidió evaluar la segmentación por color. El análisis para encontrar los valores característicos del color de la araña en la imagen siguió el mismo patrón que en la detección de brotes, es decir, analizar el histograma en los espacios de color del modelo YCbCr y segmentar los objetos presentes en la imagen. Como ejemplo se llevó a cabo este procedimiento para la figura 55. En la figura 57 se muestran las imágenes de los canales Cb (Superior Izquierda) y Cr (Inferior Izquierda) a su derecha aparecen sus respectivos histogramas.

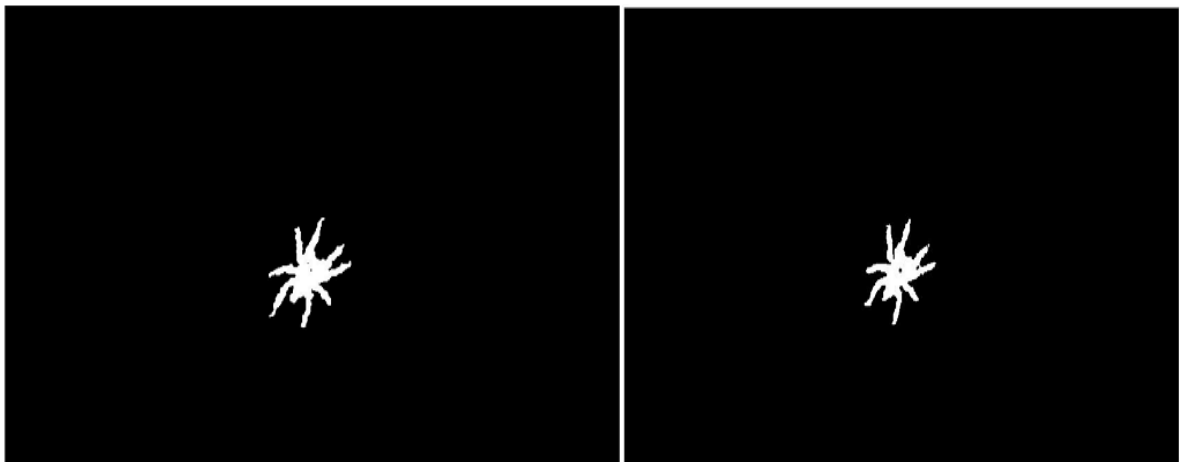
Figura 57. Ejemplo Segmentación para Fondo Negro



En los recuadros que aparecen en cada histograma están encerrados los valores equivalentes a la araña, lo que comprueba que la araña es diferenciable por color sobre el fondo negro. Estas mismas pruebas se aplicaron a varias imágenes para comprobar si los valores eran constantes, obteniendo resultados favorables. Para el canal Cb los valores de la araña oscilan entre 100 y 118, en el caso del canal Cr los valores se encuentran entre 135 y 155.

Se aplicó segmentación por umbral para ver si el contorno de la araña es apreciable, y si es posible segmentarla. En la figura 58 se exponen los resultado obtenidos tanto para el canal Cb (Izquierda) como para el canal Cr (derecha) al segmentarlos entre los valores mencionados.

Figura 58. Segmentación canales Cb y Cr para imagen da Araña



Si bien se puede apreciar que las patas de la araña quedaron un poco más cortas debido a su semejanza con el fondo en sus valores de color, la forma de la araña es diferenciable y no está fragmentada, problema presente en la segmentación en valores de gris. Se considera un resultado satisfactorio aunque, al quedar más cortas las patas de la araña los valores por relación de los descriptores iban a variar un poco, por lo que se aplicó una nueva caracterización de descriptores a imágenes de la araña sobre fondo negro, así como también para objetos probables en el área de detección, diferentes a la araña.

A continuación se muestran las gráficas para los descriptores seleccionados (Factor de Forma, Solidez y Momentos Invariantes) únicamente, ya que la variación por la segmentación afecta de igual forma a todos los objetos. Los valores en verde pertenecen a los de la araña, en rojo los valores de los objetos.

Figura 59. Gráfica Descriptor: Factor de Forma en Fondo Negro

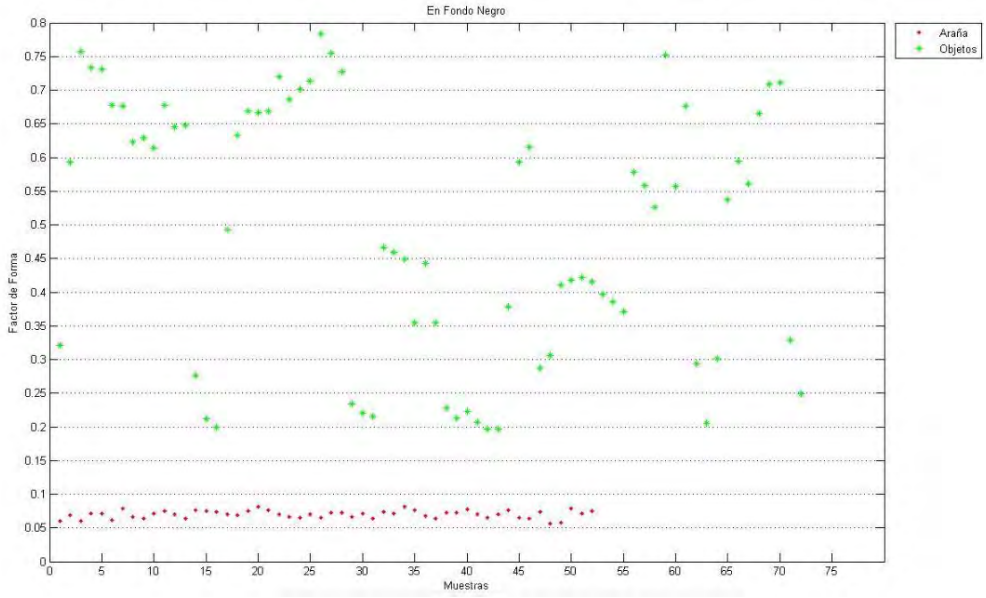


Figura 60. Gráfica Descriptor: Solidez en Fondo Negro

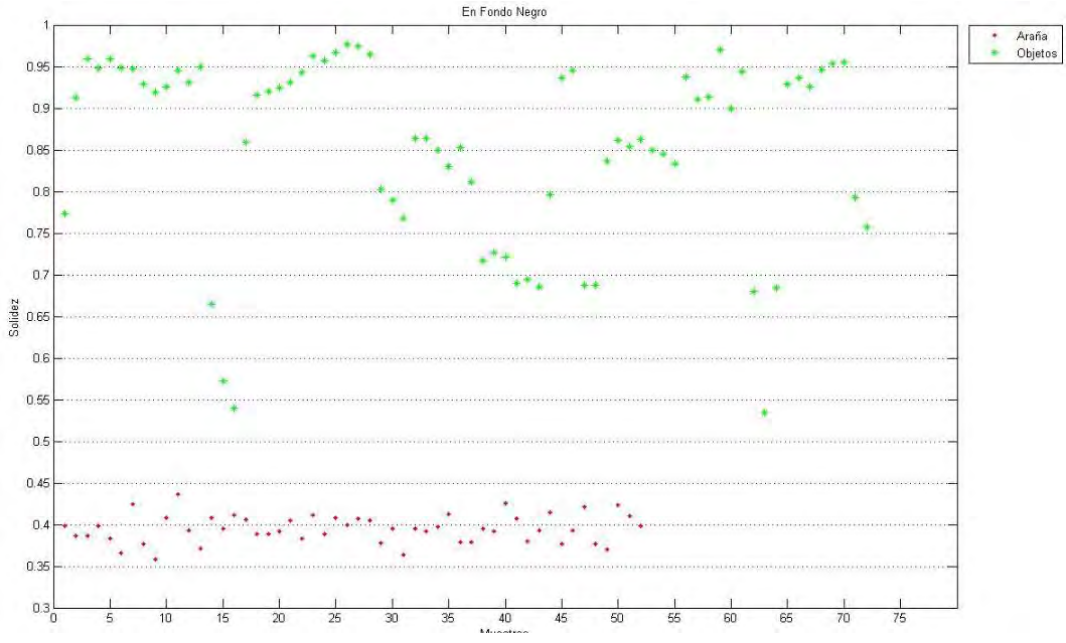
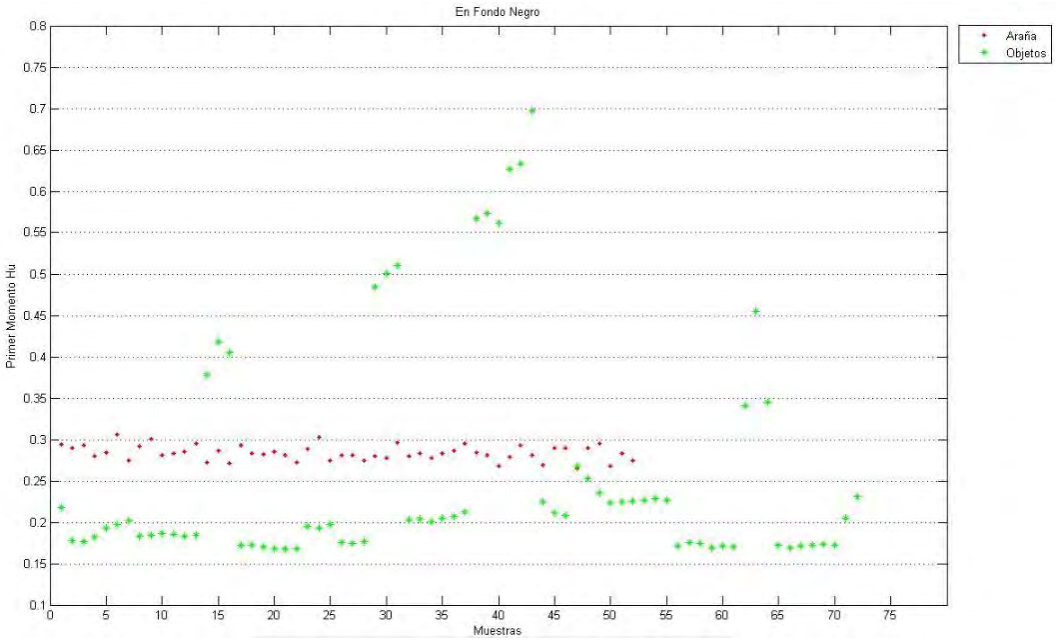


Figura 61. Gráfica Descriptor: Primer Momento Hu en Fondo Negro



Como se esperaba los valores variaron un poco, pero mantuvieron la relación de distribución, por lo que se mantuvieron los descriptores seleccionados y partir de los valores tabulados se eligieron los rangos de aceptación de detección para la araña sobre un fondo oscuro.

6.2 IMPLEMENTACIÓN EN PLATAFORMA

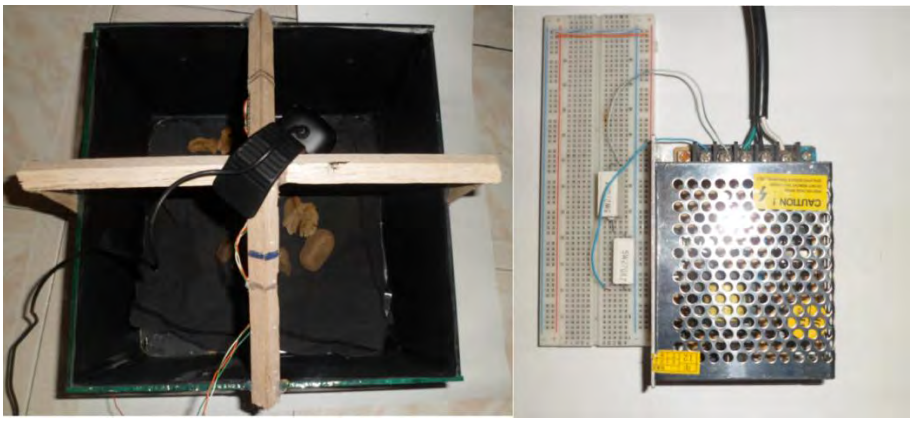
El módulo de implementación en plataforma expone los resultados obtenidos al realizar pruebas con el programa diseñado para la identificación de la araña, funcionando sobre el dispositivo de procesamiento junto con los demás componentes del sistema embebido.

Descripción Montaje para Pruebas

Para llevar a cabo las pruebas experimentales con la araña durante el proyecto, se construyó un pequeño escenario cubico hecho con paredes de vidrio, tapizados con cartulina negra, con el objetivo de simular las condiciones de oscuridad en las que puede aparecer la araña ya que tiene hábitos nocturnos. En la parte superior se colocaron unos soportes de balso para fijar la cámara web y los leds de

iluminación. Los leds están conectados a un divisor de corriente para variar su intensidad y están alimentados por una fuente regulada (12V/ 3.5A). Ver figura 62.

Figura 62. Imagen de Escenario de Pruebas y Fuente de Alimentación

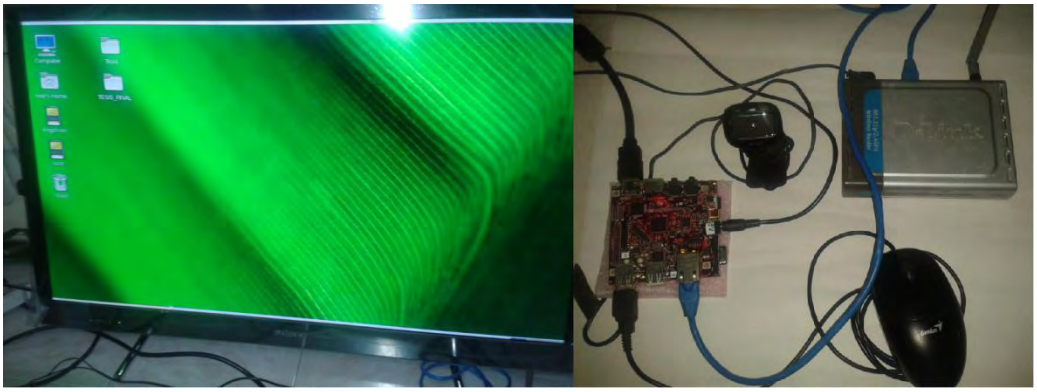


El programa diseñado para la detección es compilado y ejecutado en el dispositivo de procesamiento BB-xM, a este se conectan, la cámara y el enrutador.

Para configurar parámetros del sistema directamente en la plataforma se conectan teclado y mouse. La visualización se hace por medio de la salida HDMI que tiene la BB-xM, se puede conectar a cualquier pantalla o proyector que tenga una entrada de esta clase.

Cuando ocurre una detección el programa debe guardar las imágenes de la araña detectada para poder ser visualizada remotamente, como se muestra en el apartado de Implementación y Comunicación del capítulo 5. Para realizar la comunicación se utiliza un enrutador inalámbrico que se encarga del envío de información por WIFI.

Figura 63. Imagen de Implementación en Plataforma



Pruebas Sobre Plataforma

Para comprobar el funcionamiento del sistema se llevaron a cabo una serie de pruebas experimentales con el programa funcionando desde la plataforma, en donde se colocó la araña en la caja escenario junto con distintos objetos cómo piedras y hojas. En la tabla 1 se muestran los resultados obtenidos en una prueba hecha a 30 imágenes de la araña en distintas partes de la escena y con algunos objetos aleatorios.

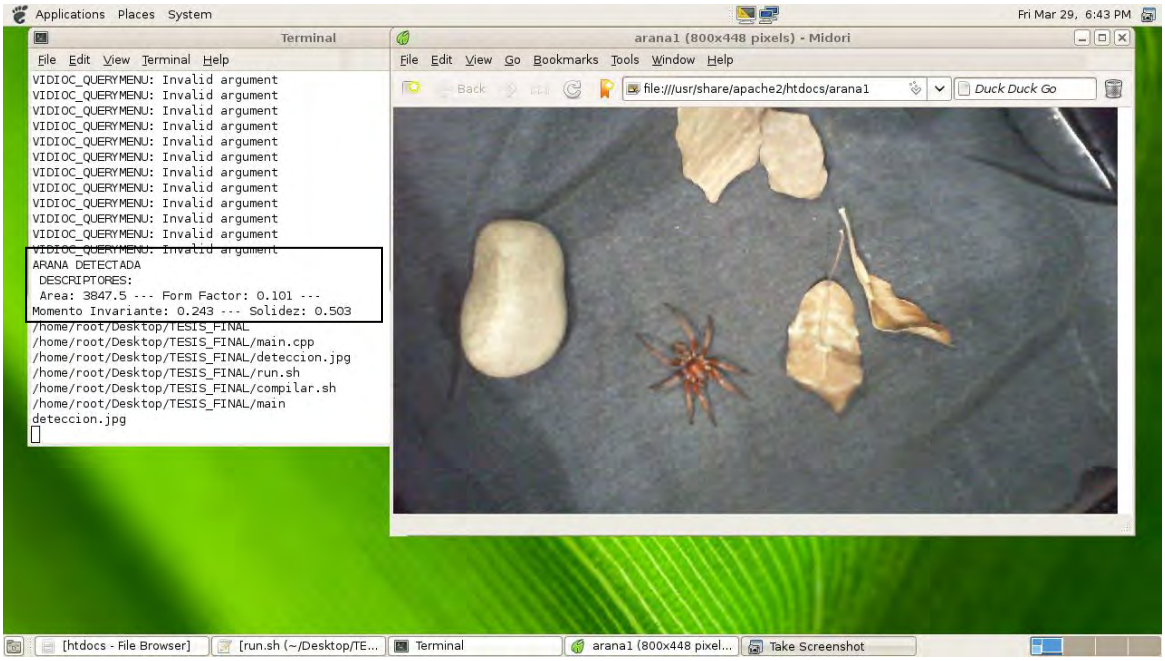
Tabla 2. Resultados de Prueba de Detección Sobre Plataforma

	Resultado
Imagen 1	P
Imagen 2	P
Imagen 3	P
Imagen 4	N
Imagen 5	P
Imagen 6	P
Imagen 7	P
Imagen 8	P
Imagen 9	P
Imagen 10	P
Imagen 11	P
Imagen 12	P
Imagen 13	P
Imagen 14	P
Imagen 15	P
Imagen 16	P
Imagen 17	P
Imagen 18	P
Imagen 19	P
Imagen 20	P
Imagen 21	N
Imagen 22	P
Imagen 23	P
Imagen 24	N
Imagen 25	P
Imagen 26	P
Imagen 27	P
Imagen 28	N
Imagen 29	P
Imagen 30	P

P=Positivos, N=Negativos

Para esta prueba de muestra se obtuvieron 26 resultados positivos y 4 negativos, lo que es un resultado satisfactorio en una muestra de 30 datos. La figura 64 muestra cómo aparecen los resultados de una detección positiva en la pantalla de visualización de la implementación.

Figura 64. Imagen de Resultado Positivo Sobre Plataforma



Después de que sucede una detección exitosa, la imagen que queda almacenada en el servidor, puede ser vista de forma remota, conectándose en la misma red en la que está el sistema con cualquier dispositivo que tenga WIFI y al que se le pueda establecer una puerta de enlace predeterminada. En la figura 65 se muestra la página HTML donde se pueden ver las imágenes con detecciones, esta fue tomada desde un PC de forma inalámbrica por medio de la conexión WIFI.

Figura 65. Imagen de Araña detectada Vista en Pagina HTML

7. COSTOS

Las tablas presentadas a continuación, muestra los costos de desarrollo del proyecto, donde se incluyen costos de adquisición del hardware y los materiales necesarios para el estudio y desarrollo del sistema de visión Artificial.

Tabla 3. Costo de Hardware

HARDWARE DEL SISTEMA DE VISIÓN ARTIFICIAL			
CANT	**ELEMENTO**	**COSTO**	
1	Beagle Board xM	$	320.000
1	Teclado USB	$	20.000
1	Mouse	$	10.000
1	Micro SD 4GB	$	16.000
1	Cable de comunicación HDMI	$	15.000
1	Cámara HD 300	$	130.000
1	Router Inalámbrico D-Link Dl-524	$	135.000
1	Cable Ethernet RJ-45	$	3.500
TOTAL		**$**	**649.500**

Tabla 4. Costo de Materiales

MATERIALES			
CANT	**ELEMENTO**	**COSTO**	
1	Fuente regulada 12V/3A	$	35.000
2	Led 5W	$	55.000
1	Balso	$	2.500
1	Cinta Negra	$	2.200
1	Caja de Vidrio de 30cm X 30cm	$	35.000
1	Cartulina Negra X Pliego	$	2.500
1	Silicona	$	1.500
TOTAL		**$**	**133.700**

Tabla 5. Costo de Adquisición de Especies

ADQUISICIÓN DE ESPECIES			
CANT	**ELEMENTO**	**COSTO**	
2	Arañas Lycosidaes	$	40.000
2	Arañas Ctenidae	$	60.000
1	Estudio de Información Taxonómica Arácnidos	$	40.000
4	Hábitat para Arañas	$	60.000
TOTAL		**$**	**200.000**

Tablas 6. Costo Total del Proyecto

ADQUISICIÓN DE ESPECIES			
CANT	**ELEMENTO**	**COSTO**	
1	Hardware del sistema de visión artificial	$	649.500
1	Materiales	$	133.700
1	Adquisición de especies	$	200.000
TOTAL		**$**	**983.200**

8. CONCLUSIONES

La investigación realizada para establecer cuál era la especie de arácnido a ser identificada por el sistema de visión artificial propuesto, indicó que las especies con mayor probabilidad de aparición en una zona de pos fuego en la región geográfica donde se encuentra el jardín botánico de Cali, son las arañas pertenecientes a las familias Lycosidae y Ctenidae. La información recabada de estas dos familias, la Lycosidae es la que tiene un mayor porcentaje de aparición en las primeras etapas de pos fuego, con un 48% comparada con otras familias. De las arañas de este tipo extraídas de la zona de detección, el tamaño de las Lycosidaes encontradas oscilaba entre los 2cm y 4cm, lo que es un tamaño adecuado para que el sistema de visión artificial detecte satisfactoriamente.

Una vez determinada la familia de arañas a detectar, se deben extraer los valores numéricos de las características que la describen y que a la vez la diferencian de otros objetos presentes en el posible escenario de detección. Estos valores característicos o descriptores pueden representar datos geométricos, de región, color entre otros. La extracción de los valores descriptivos se aplicó a una base de datos de imágenes de arañas y de objetos aleatorios probablemente presentes en la zona de detección para compararlos, esto se hizo utilizando funciones existentes en la librería de visión artificial OpenCV. Los resultados de las pruebas mostraron que los descriptores más apropiados para identificar a la araña por su dispersión comparada con la de los demás objetos, fueron el Factor de Forma, la solidez y el primer momento invariante Hu.

Para realizar la identificación de la especie de la araña y brotes de flora, se investigaron los procesos que componen los sistemas de visión artificial, para ajustarlos y aplicarlos en este proyecto. Los algoritmos utilizados en la segmentación e identificación del brote de flora se basaron en el estudio de modelos de color digital para aprovechar el contraste entre el color natural de la planta y el fondo más oscuro afectado por el incendio. El modelo de color seleccionado fue el YCbCr ya que después de los experimentos realizados, este fue el que presento mejores resultados de segmentación, comparado con los demás modelos de color evaluados.

En el procedimiento para la segmentación y detección de la araña, se evaluaron distintos métodos de segmentación por umbralización, los cuales mostraron ser muy eficientes en fondos claros donde la araña contrastaba con el fondo. Al aplicarlos a las imágenes de la araña sobre fondo oscuro, no se consiguió hacer la segmentación satisfactoriamente, por lo que se evaluó la segmentación por color previamente utilizada para segmentar los brotes de flora. Este tipo de

segmentación permite extraer el contorno de la araña del fondo oscuro, haciendo posible el análisis de descriptores.

El criterio principal para el diseño del sistema embebido fue la capacidad de integrar el procesamiento de los algoritmos con algún sistema operativo en el cual fuera posible la instalación de las librerías optimizadas de visión computacional OpenCV y que permitiera la integración de módulos periféricos como la cámara o el enrutador inalámbrico. La evaluación de dispositivos con estas características empezó con la FPGA Cyclone II, técnicamente capaz de realizar sin problemas los procesos requeridos, pero con varios inconvenientes relacionados con la poca información disponible sobre aplicaciones de este tipo y la necesidad de crear drivers para los periféricos y funciones para la construcción del programa con los algoritmos de detección. Después se evaluó la mini6410, otro sistema capaz de realizar los procesos requeridos, la posibilidad de instalar sistemas operativos conocidos como Windows y Linux, hizo más sencilla la tarea de manejo de periféricos por los drivers ya existentes. El inconveniente que presentó esta plataforma al igual que la FPGA fue que no había información relacionada con esta y no fue posible integrar las librerías de OpenCV al kernel del sistema operativo, por lo que cualquier intento realizado para implementar en esta tarjeta, no funcionó.

La tercera plataforma evaluada fue la BeagleBoard-xM, que resulta ser muy utilizada en proyectos con sistemas de visión artificial y procesamiento de imágenes. Sobre este tipo de proyectos implementados en la BB-xM hay buena cantidad de información disponible en internet. Se le pueden instalar distintos sistemas operativos y conectar distintos periféricos al igual que la mini6410, con la diferencia de que tiene muchas facilidades para incluir distintos tipos de librerías en el kernel del sistema operativo, incluidas las librerías de OpenCV. Por esto se decidió trabajar con la BeagleBoard-xM como módulo de procesamiento para el sistema de detección. La migración de los algoritmos diseñados en Windows al sistema operativo Linux con distribución Angstrom instalado en la BB-xM, fue sencilla ya que tomo pocos pasos configurar el embebido para que el código funcionara de manera óptima en la plataforma.

La iluminación es un factor muy influyente en los sistemas de visión artificial, aun tratando de disminuir su influencia con el preprocesamiento en las imágenes, las variaciones de esta puede afectar drásticamente procedimientos como la segmentación. Las pruebas de validación de funcionamiento realizada sobre el sistema embebido, consistió en poner en marcha la rutina de detección y tomar 30 muestras con escenarios modificados con la araña. Se obtuvieron 26 aciertos en los que la araña fue identificada satisfactoriamente. Los 4 resultados negativos están relacionados con las limitaciones presentes en la segmentación por color, en

la cual es difícil diferenciar cuando la araña está sobre algún objeto que tenga valores de color cercanos al suyo, provocando que el sistema lo tome como un solo objeto.

9. TRABAJO FUTURO

Este proyecto tiene como meta a futuro desarrollar una base de datos con los descriptores de diferentes tipos de insectos que permitan al sistema clasificar e identificar de forma automática los diferentes géneros de fauna en un ecosistema determinado.

Para mejorar la calidad de la imagen se propone una cámara con sensor CMOS de alta calidad que permita enfocar y resaltar mejor las características físicas del insecto, sobre todo para insectos de tamaño inferior a 1 cm como son las hormigas y algunos tipos de arácnidos.

Para optimizar el sistema de iluminación de la plataforma se propone un sistema de auto iluminación que permita aumentar la intensidad de brillo cuando se detecte imágenes opacas, tal como funcionan las cámaras actuales que tienen corrección de iluminación o autofoco. Esto permitiría tener una mejor calidad de imagen lo que significa, códigos menos robustos para el preprocesamiento haciendo el sistema más robusto y rápido.

BIBLIOGRAFÍA

AN,Lu; XINWEN, Ho; XIAOLIN, Chen y CHIENGLIN, Liu. Institute of Automation Chinese Academy of Sciences. Insect Species Recognition using Sparse Representation. Beijing. 2010 [Citado en 15 Abril 2012]

BRADSKI, Gary. KAEBLER, Adrian. Learning OpenCV – Computer Vision with the OpenCV Library. O'reilly. USA, 2008. 577 p.

CABREJAS F, Héctor. Detección y eliminación de sombras y reflejos en entornos de video – seguridad sobre plataforma de analistas distribuido. Madrid - España: Universidad Autónoma de Madrid, Escuela Politécnica Superior, 2010. 138 p.

CYPRESS PERFORM. PSOC PROGRAMMABLE SYSTEM-ON-CHIP. [Online]. [Citado 10 Marzo 2012]. Disponible en: http://www.cypress.com/?docID=32416&dlm=1

DO, HARP y NORRIS. Departmen of Computer Sciences, University of Tenessee. A test of a pattern recognition system for identification of spiders. Knoxville. 1999 [Citado en 14 Abril 2012]

FLÓREZ López Raquel, José M. Fernández Fernández. Las Redes Neuronales Artificiales. p. 12

FRANK Y. SHI. Image Processing and Pattern Recognition, Fundamentals and Techniques. IEEE Press, p. 40

GONZALEZ Rafael C., Richard E. Woods. Digital Image Processing, Third Edition. Addison – Wesley Iberoamerican. S.A. [1996]. ISBN 0-201-62576-8. p. 632

HUIDOBRO, José. Sistemas telemáticos. Madrid. International Thomson ediciones, 2007. p. 242

I. T. L. Education Solutions Limited. Introduction to Computer Science. 2 ed. Delhi. Pearson Education, 2011. pp. 96-97

ISAAC N. Bankman, PhD, handbook of medical imaging processing and analysis, Editor in Chief, Academic Press, p 69-195.

KAMAL, Raj. Microcontrollers: Architecture, Programming, Interfacing and System Design. 1 ed. New Delhi.: Prentice Hall, 2009. p. 3

KUON, Ian; TESSIER, Russell y ROSE, Jonathan. FPGA Architecture. Hanover. Now Publishers Inc, 2008. p. 1

LÓPEZ, Jesús y CAICEDO, Eduardo. Una Aproximación Práctica a las Redes Neuronales Artificiales. Editorial Universidad del Valle 2009 , p9

MARCOS GONZÁLEZ, Ana, integrantes del grupo de investigación EDMANS. Universidad de La Rioja. Técnicas y algoritmos básicos de visión artificial. Servicios de publicaciones, 2006. ISBN 84-689-9345-X, p. 19

NAKAMURA, Junichi. Image Sensors and Signal Processing for Digital Still Cameras. Boca Raton.: Taylor & Francis Group, LLC, 2006. p. 2

OSHANA, Robert. DSP Software Development Techniques for Embedded and Real-Time Systems. Newnes, 2005. p. 11

UJALDÓN, Manuel. Arquitectura del PC, Volumen 3. Madrid. CIENCIA 3. DISTRIBUCIÓN, S.A, 2003. p. 37

VAZQUEZ J.F, LUNA C.A. Umbral adaptativo para la detección de objetos en movimiento usando visión computacional, Departamento de Telecomunicaciones, Universidad de Oriente Santiago de Cuba, Cuba, AAEI 05, Santander 28-30 Sept.

VERA, Pablo; SALAS, Joaquín; MANDUCHI, Roberto y SANCHEZ, Emiliano. CICATA-IPN .Detección de Flores mediante el Análisis de Secuencias de Imágenes. Queretaro. Mayo, 2009 [Citado 14 Abril 2012].

ZULOAGA, Aitzol y Astarloa, Armando. Sistemas de Procesamiento Digital. 1 ed. Madrid. Delta Publicaciones, 2008. P. 252

ANEXOS

Anexo A. Guía de Instalación de OpenCV y Code Blocks

Instalación de OpenCV 2.4.2 y Code Blocks 10.05

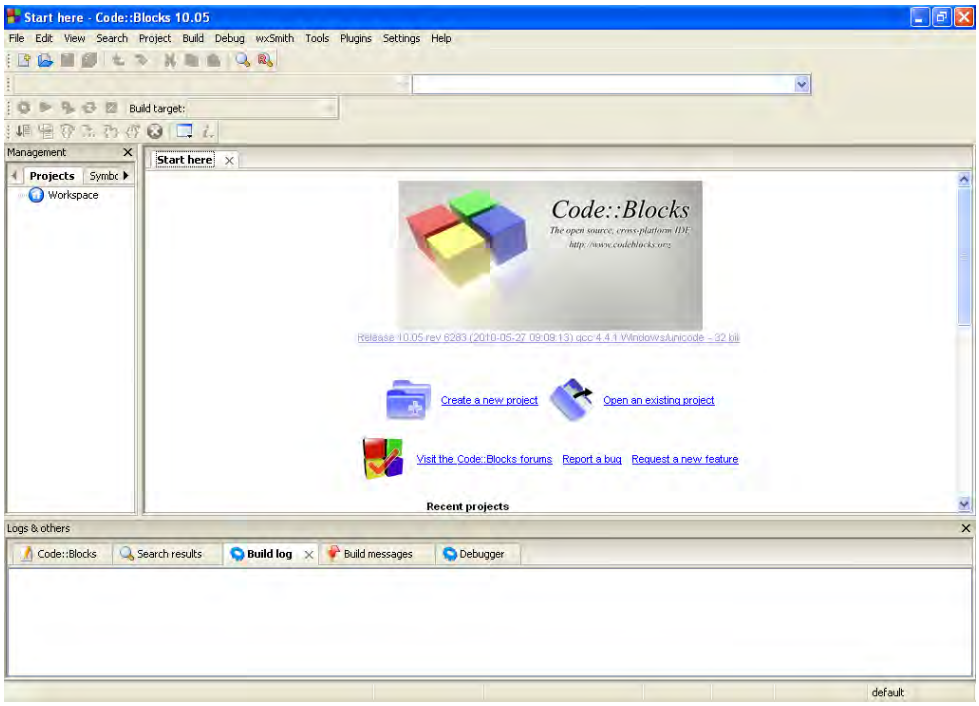
Pasos:

1. Descargar Code Blocks 10.05, ingresar a la página principal http://www.codeblocks.org, MAIN clic Downloads, Binaries, clic en codeblocks-10.05mingw-setup.exe, descargar por Sourceforge.net**.**

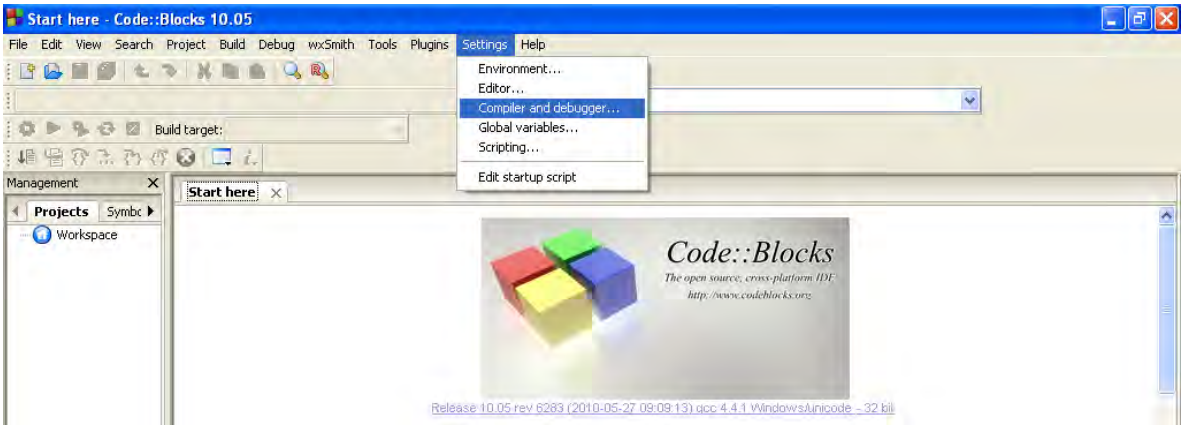
2. Ejecutar el .EXE, una vez instalado comprobar que Code Blocks está funcionando, debe salir un mensaje que dice que ha detectado el compilador GNU Gcc.

3. Descargar OpenCV 2.4.2, ingresar a http://sourceforge.net/projects/opencvlibrary/files/opencv-win/2.4.2/, descargar el archivo .EXE, descomprimir en Disco Local (C:)

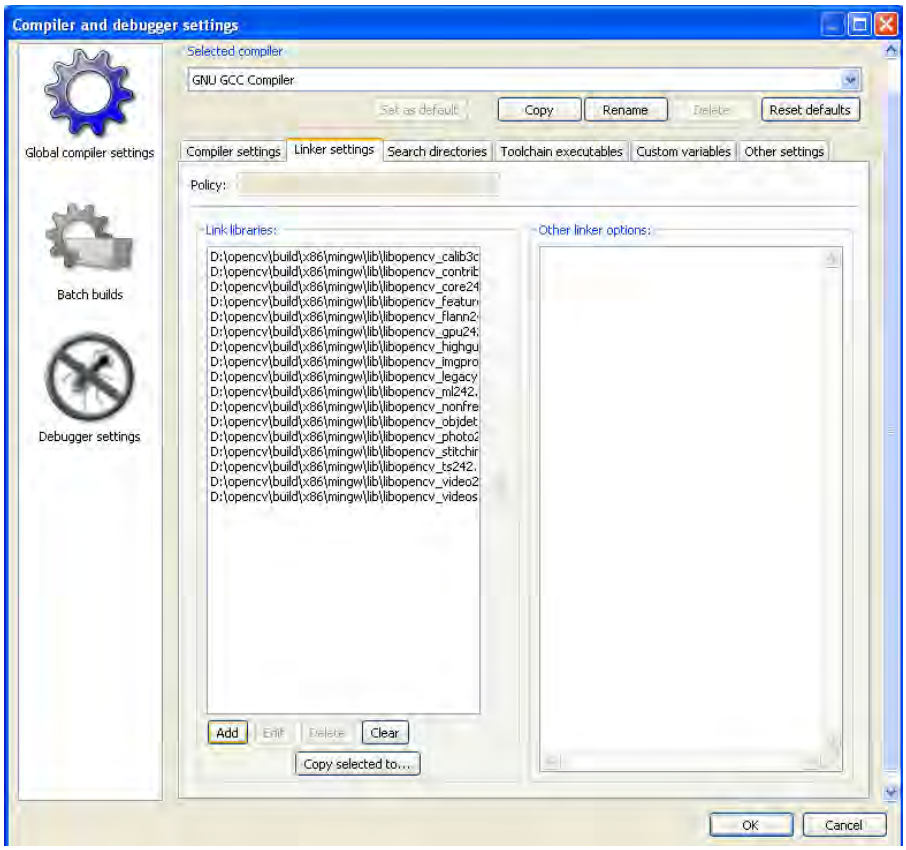
4. Abrir Code Blocks 10.05.



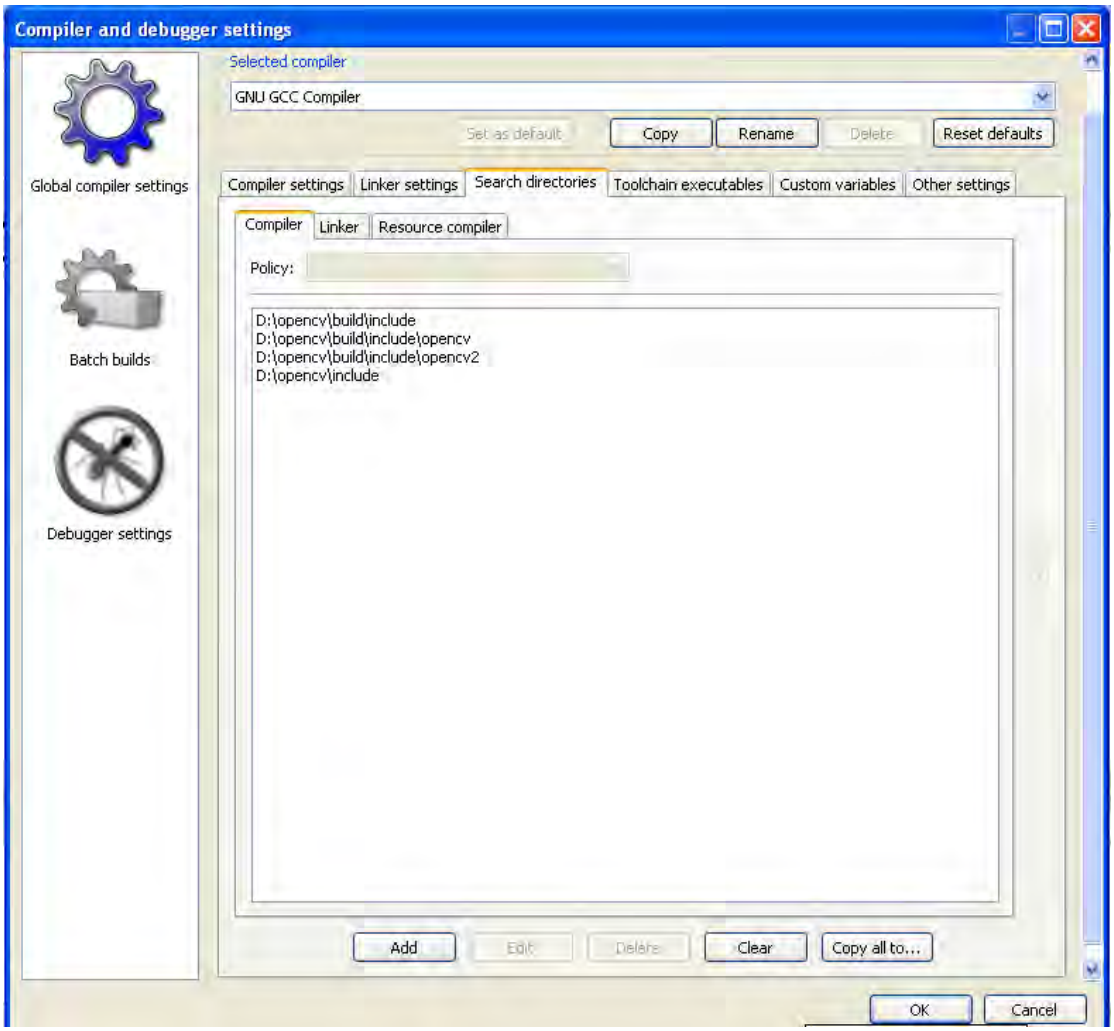
5. Agregar las librerías de OpenCV, para esto dar clic en *Compiler And Debugger*… como indica la figura.



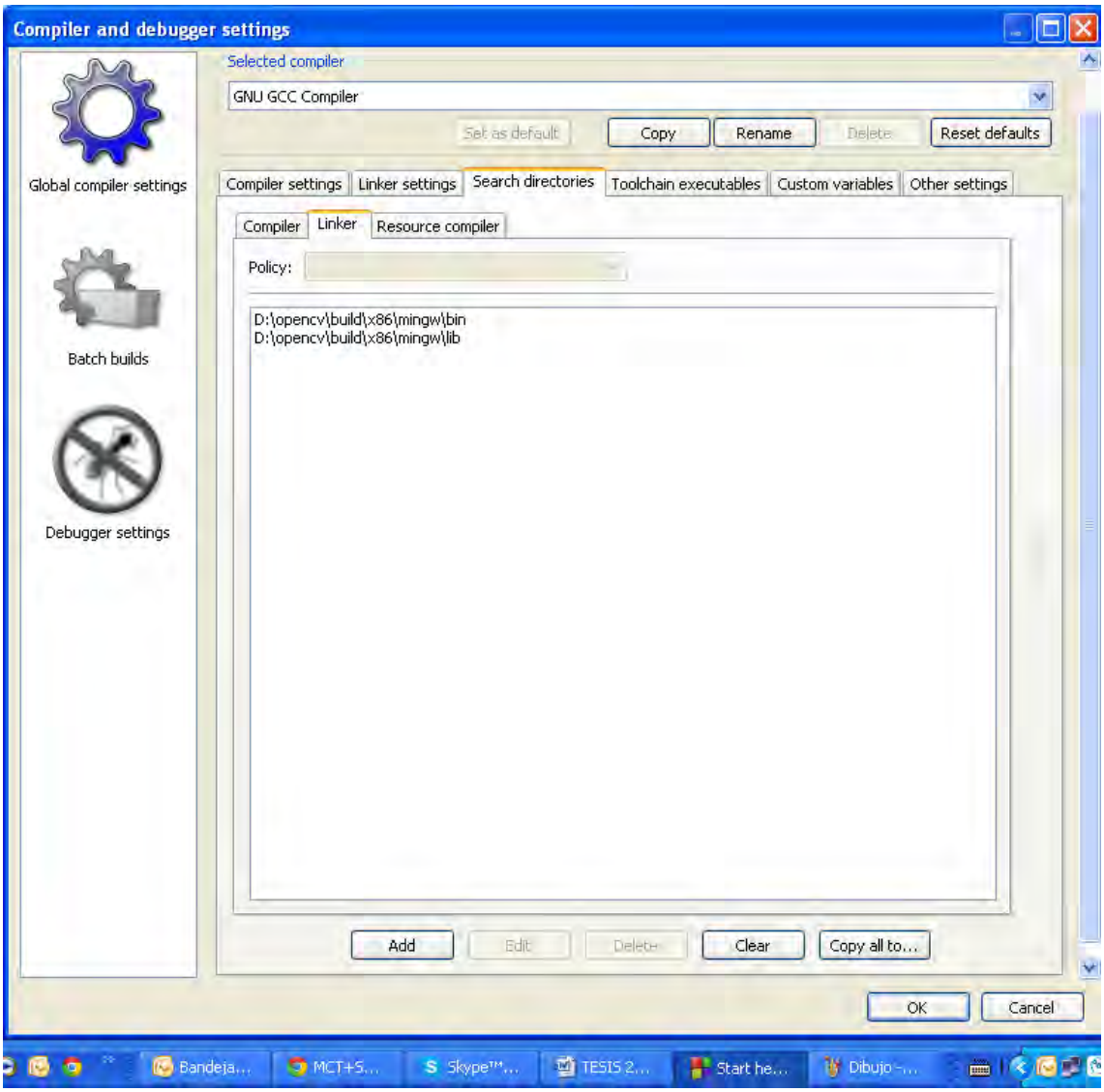
6. Clic en *Linker Settings,* luego agregar todas las librerías que vienen en el super pack de OpenCV, D:\opencv\build\x86\mingw\lib.



7. Clic en *Search Directories*, clic en Compiler > Agregar cada una de las carpetas "include" de los módulos de OpenCV y la carpeta "include" de OpenCV, como se muestra:



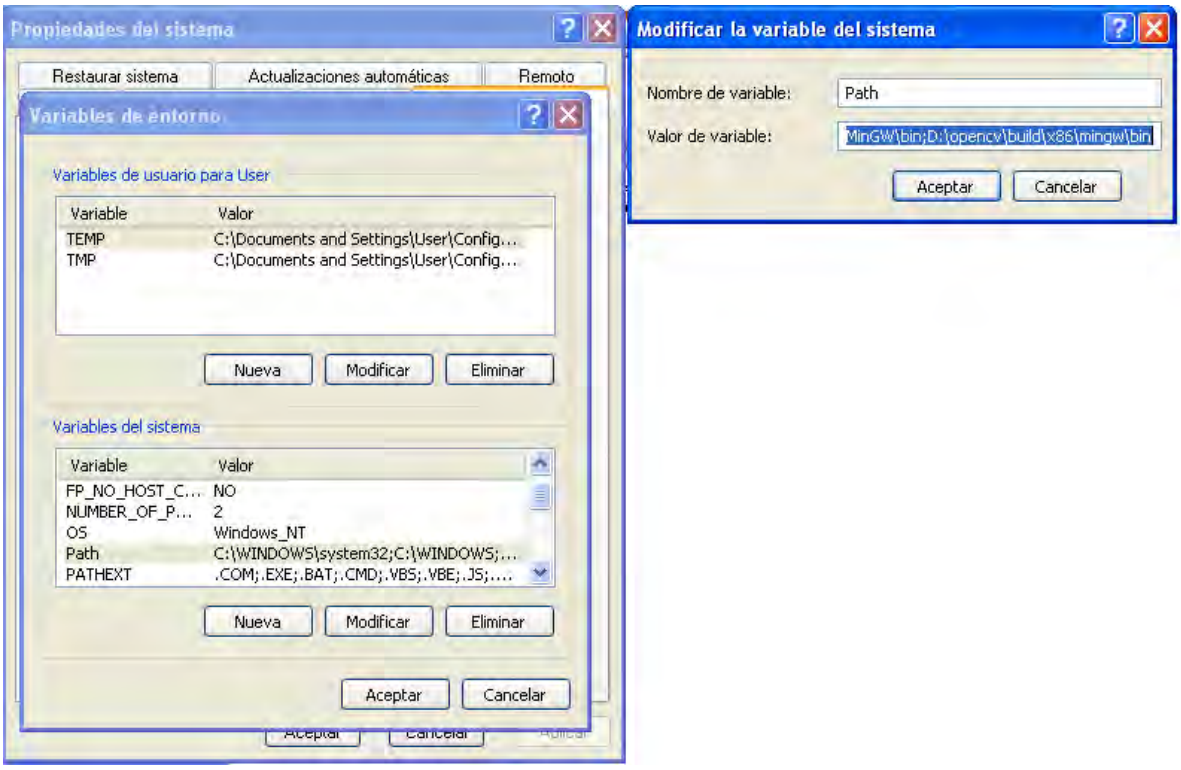
8. Clic en *Search Directories*, clic en Linker > Agregar la carpeta "bin" y "lib", como se muestra en la figura.



9. Agregar los Paths de Code Blocks y OpenCV, ir a panel de control, Sistema, Opciones Avanzadas, Variables de Entorno, clic en Path y agregar las direcciones de los “bin”.

- C:\Archivos de programa\CodeBlocks\MinGW\bin;
- D:\opencv\build\x86\mingw\bin

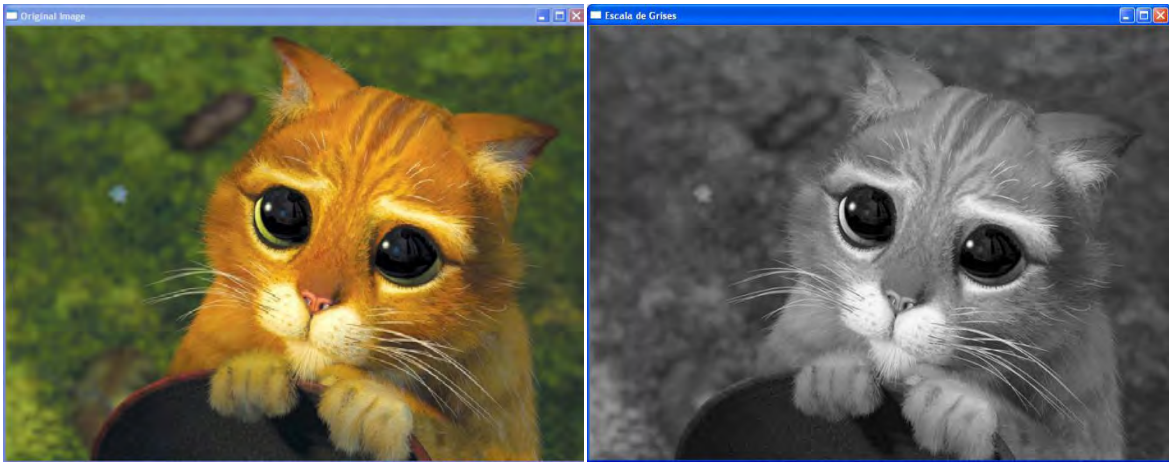
Como se muestra en la figura.



10. Se realiza un código de prueba:

```
#include "opencv2/highgui/highgui.hpp"
#include "opencv2/imgproc/imgproc.hpp"
#include <cv.h>
#include <iostream>
#include <stdio.h>
#include <stdlib.h>

#define WIDTH 640 // Columns
#define HEIGHT 480 // Rows

using namespace cv;
using namespace std;
IplImage* img = cvLoadImage("C:/Documents and Settings/User/Escritorio/gato.jpg");

int main()
{
	IplImage* resultado = NULL;
	resultado = cvCreateImage(cvGetSize(img), 8, 1);

	cvNamedWindow("Original Image",CV_WINDOW_AUTOSIZE);
	cvShowImage("Original Image", img);

	cvCvtColor(img,resultado,CV_BGR2GRAY);

	cvNamedWindow("Escala de Grises",CV_WINDOW_AUTOSIZE);
```

```
        cvShowImage("Escala de Grises", resultado);

        cvWaitKey(0);
        cvDestroyWindow("CONTORNO");
        cvReleaseImage(&img);
        cvReleaseImage(&resultado);

        return 0;
}
```



Anexo B. Guía para integrar OpenCV con el SO Angstrom

Configuración de la plataforma Beagle Board para usar OpenCV.

1. lo primero que hay que hacer es formatear la memoria SD de tal manera que pueda arrancar el SO. Para esto se necesita tener alguna distribución de Linux instalada en el computador donde va a dar formato a la memoria SD, para esta guía se trabajó con la distribución de Ubuntu 10.04x32.

2. Insertar la SD en la máquina, para saber si la memoria está montada ejecutar el siguiente comando en el terminal.

```
df -h
```

Usted verá algo como esto:

```
/dev/sda5               98G   56G   38G  61% /
none                   1.5G  316K  1.5G   1% /dev
none                   1.5G  724K  1.5G   1% /dev/shm
none                   1.5G  336K  1.5G   1% /var/run
none                   1.5G     0  1.5G   0% /var/lock
none                   1.5G     0  1.5G   0% /lib/init/rw
none                    98G   56G   38G  61%
/var/lib/ureadahead/debugfs
/dev/sdb1               15G  8.0K   15G   1% /media/FAE3-DCE5
```

La SD card es /dev/sdbx, verificar el tamaño de la memoria, en este caso sdb1 corresponde a la primera partición de la memoria.

3. Para que Angstrom se ejecute, necesita dos particiones, una partición donde va aguardar los Boot Files y una tipo ext3 donde almacenara los Root Files del sistema.

Existen dos formas de configurar las particiones, una es de forma manual utilizando el program fdisk de Ubuntu los pasos para la configuración se encuentran disponibles en el siguiente link:

http://opencvjaveriana.wikispaces.com/configuracion_plataformas+de+procesamiento.

El otro método es generando las dos particiones de forma automática con un Script diseñado por Graeme Gregory. La fuente del programa de configuración fue tomado de:

- http://treyweaver.blogspot.com/2010/10/installing-angstrom-on-beagleboard-xm.html

Para este último solo se necesita ejecutar el siguiente comando en el terminal:

```
mkdir ~/angstrom-wrk
cd ~/angstrom-wrk
wget
http://cgit.openembedded.org/cgit.cgi/openembedded/plain/contrib/angstrom/omap3-mkcard.sh
chmod +x omap3-mkcard.sh
sudo ./omap3-mkcard.sh /dev/sdbX
sync
```

Tener en cuenta el nombre con el cual aparece montada la memoria /dev/sdbX.

4. Remover la SD card, esperara tres segundo para volver a introducirla, verificar que la memoria está montado con el comando:

```
df -h
```

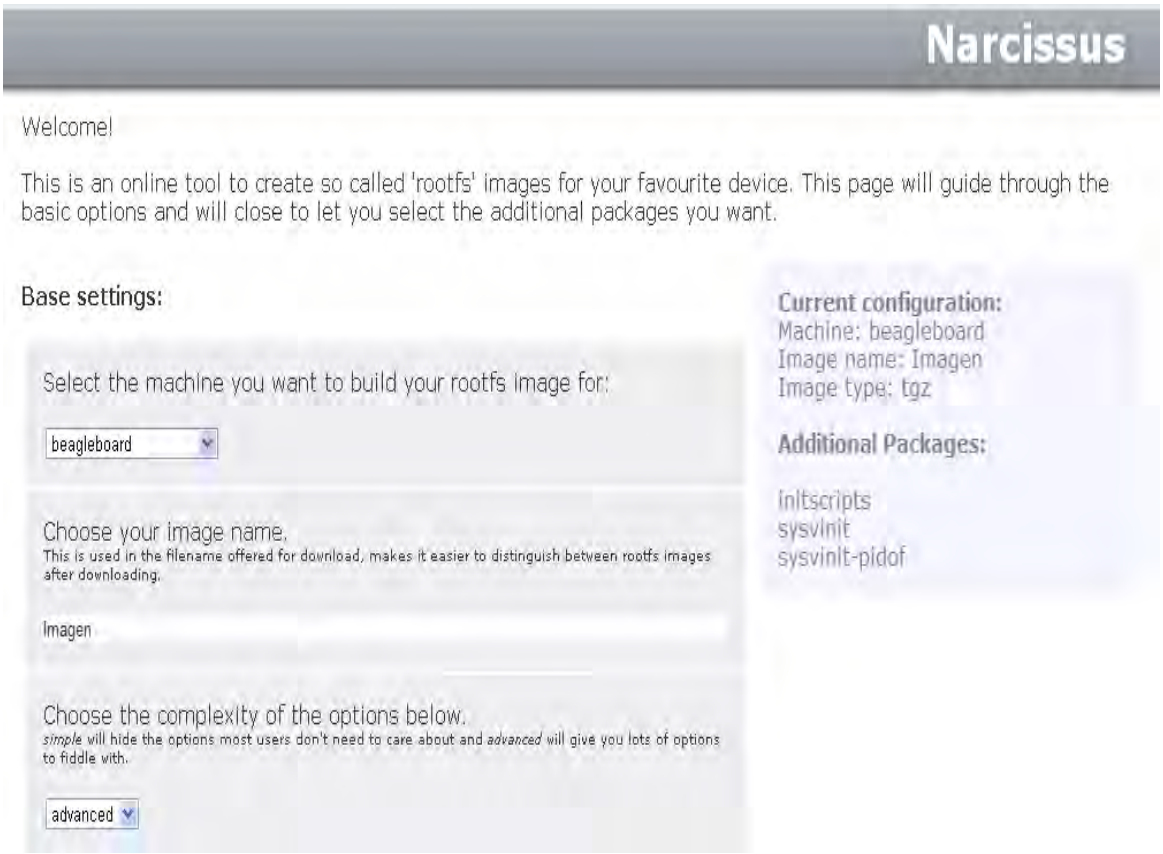
Ahora usted tiene dos nuevas particiones:

```
/media/boot
/media/Angstrom
```

5. Para poder compilar el Kernel de Angstrom con las librerías de OpenCV, debe seguir los siguientes pasos:

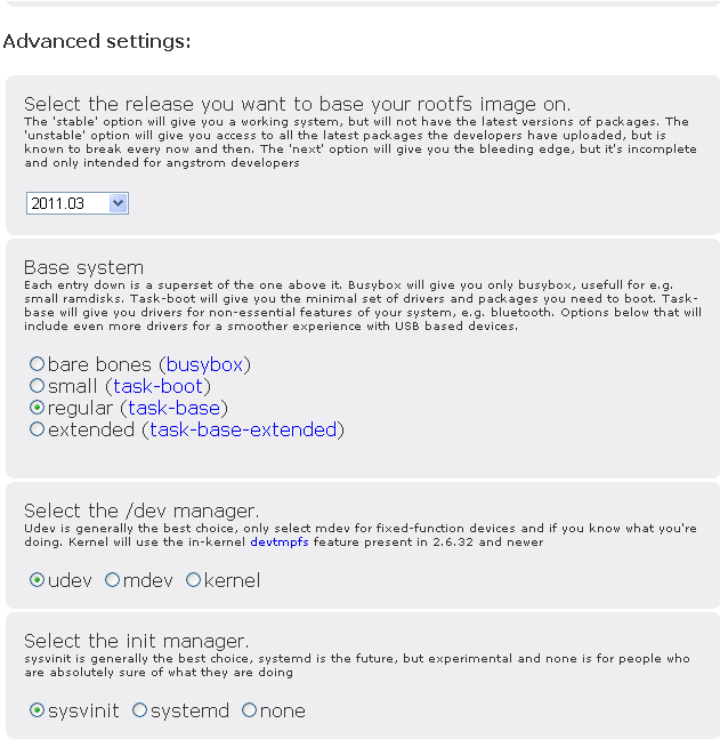
   - Ir http://narcissus.angstrom-distribution.org/
   - Seleccione:
     - **Machine:** beagleboard
     - **Image name:** Escoger Nombre.
     - **Complexity:** advanced

   Ver Imagen.

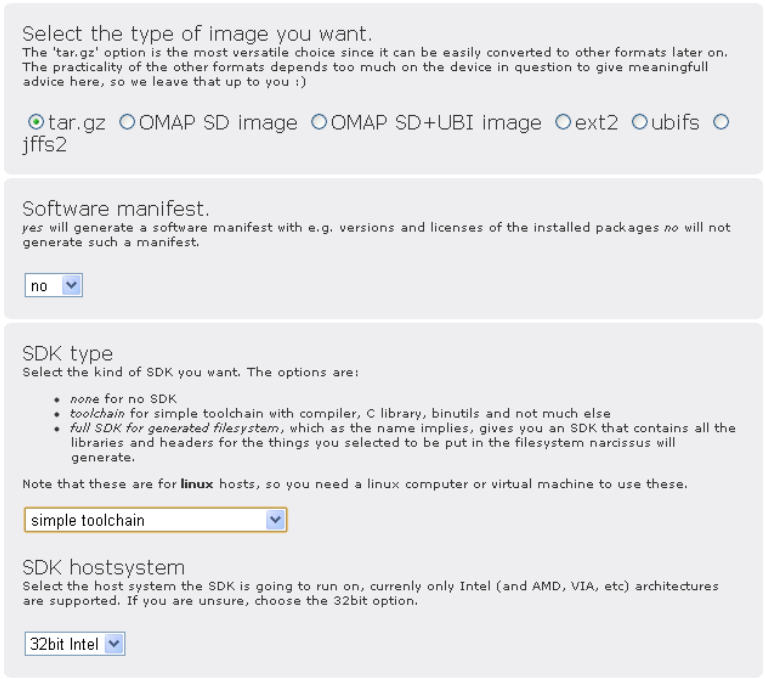


   - **Release (default):** 2011.03
   - **Base system (default):** regular (task-base)
   - **/dev manager (default):** udev
   - **Init Manager:** Sysvinit

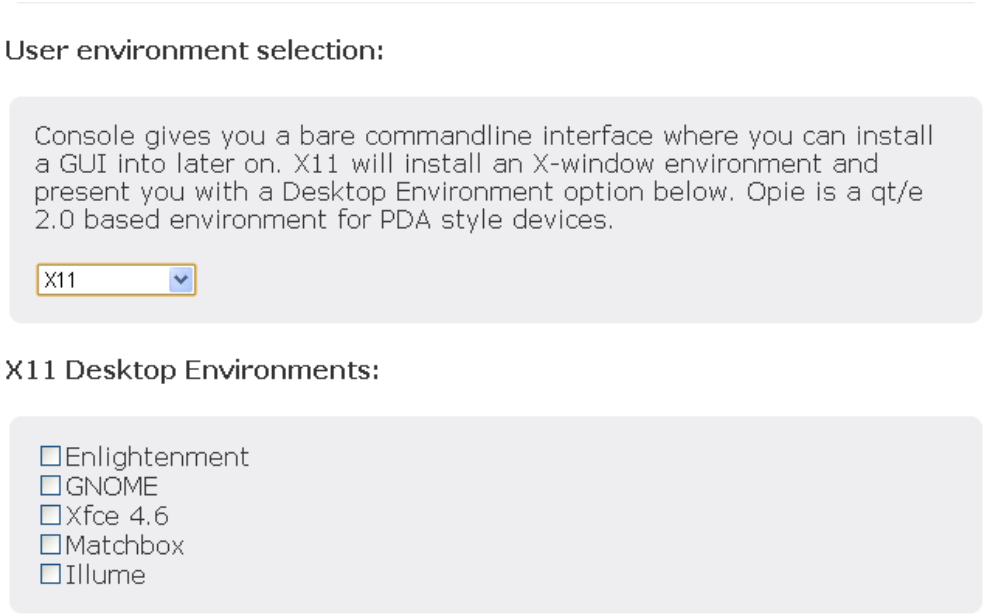
   Ver Imagen:



- **Type of image:** tar.gz
- **Software manifest:** no
- **SDK type:** simple toolchain
- **SDK hostsystem (default):** 32bit Intel



- **User environment selection:** X11
- **X11 desktop environment:** Dejar todos en blanco.



Additional packages selection:

- **Additional X11 packages:** Seleccionar Firefox, Firefox mediaplayer plugin, Midori web browser.

- **Development packages:** Seleccionar OpenCV headers and libs

- **Additional console packages:** Seleccionar All kernel modules, Nano Editor, OpenCV

- **Network related packages:** Seleccionar Apache, NetworkManager, NetworkManager GUI applet, Wireless-tools.

- **Platform specific packages:** Seleccionar OMAP Display Sub System (DSS) , BeagleBoard validation GUI , BeagleBoard validation GNOME image.

6. Clic en “Built me!” y esperar el tiempo requerido para que se construya el Kernel con los paquetes y librerías que se seleccionaron.

7. Descargue todas las Imágenes.

8. Extraer los archivos del directorio Boot de la imagen:

```
tar --wildcards -xjvf [YOUR-DOWNLOAD-FILE].tar.bz2 ./boot/*
```

9. Copiar MLO, uImage, U-boot.bin a la partición /media/boot de la SD Card.

```
cp boot/MLO-* /media/boot/MLO
cp boot/uImage-* /media/boot/uImage
p boot/u-boot-*.bien /media/boot/u-boot.bin
sync
```

10. Extraer y copiar los Root Files de la Imagen a la partición /media/Angstrom de la SD card.

```
sudo tar -xvj -C /media/Angstrom -f [YOUR-DOWNLOAD-
FILE].tar.bz2
sync
```

11. Remueva la SD card espere unos segundos y compruebe que los archivos copiados estén en las particiones.

```
sync
umount /media/boot
umount /media/Angstrom
```

12. Por último, inserte la SD card a la Beagle Board y espere hasta que finalice la instalación, puede tardar varios minutos. Una vez finalice la instalación se inicia el sistema operativo Angstrom ver imagen del escritorio.

13. Ya instalado el sistema operativo con las librerías de OpenCV, ahora se procede a actualizar algunos paquetes y librerías. Ingrese al Terminal y ejecute los siguientes comandos:

```
opkg update
opkg install task-native-sdk cpp gccmakedep
opkg install python-distutils python-compile python-compiler
python-devel
opkg install ffmpeg-dev
opkg install opencv
```

14. Se realiza un Script que permita compilar los algoritmos hechos en Windows con las librerías de OpenCV, esto genera un ejecutable que corre la aplicación. Tomado de OpenCV Javeriana. Tema Instalar OpenCV en Angstrom.

    http://opencvjaveriana.wikispaces.com/configuracion_plataformas+de+procesamiento.

```
#!/bien/su
echo "compilando..."
```

```
export LD_LIBRARY_PATH=/usr/local/lib
echo `pkg-config --cflags opencv`
echo `pkg-config --libs opencv`
g++ `pkg-config --cflags opencv` -ggdb main.cpp `pkg-config --
libs opencv`
echo "termina de compilar"
```

15. Para ejecutar la aplicación se puede hacer doble clic sobre el ejecutable “main” o desde el terminal.

Anexo C. Guía de configuración servicio HTTP Apache

Configuración de Apache la plataforma Beagle Board.

1. Configurar una dirección IP estática, para esto ir a la carpeta network

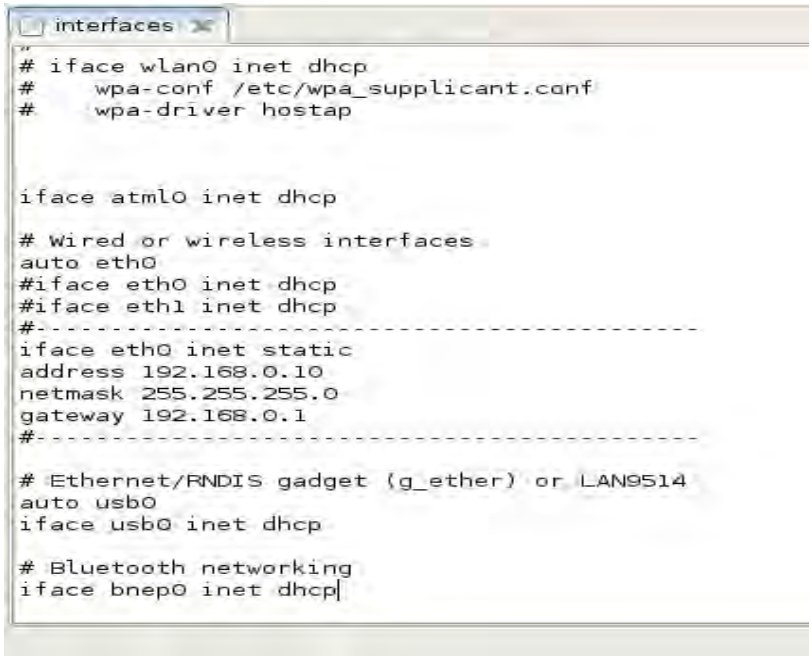
2. Desde la Consola del terminal de Angstrom cd /etc/network

3. Dentro de la carpeta gedit /interfaces

4. Comentar la línea iface eth0 inet dhcp y iface eth1 inet dhcp

5. Escribir la siguiente línea:

   iface eth0 inet static
   Adress: xxxx.xxxx.xxxx.xxx
   Netmask: xxxx.xxxx.xxxx.xxx
   Gateway: xxxx.xxxx.xxxx.xxx



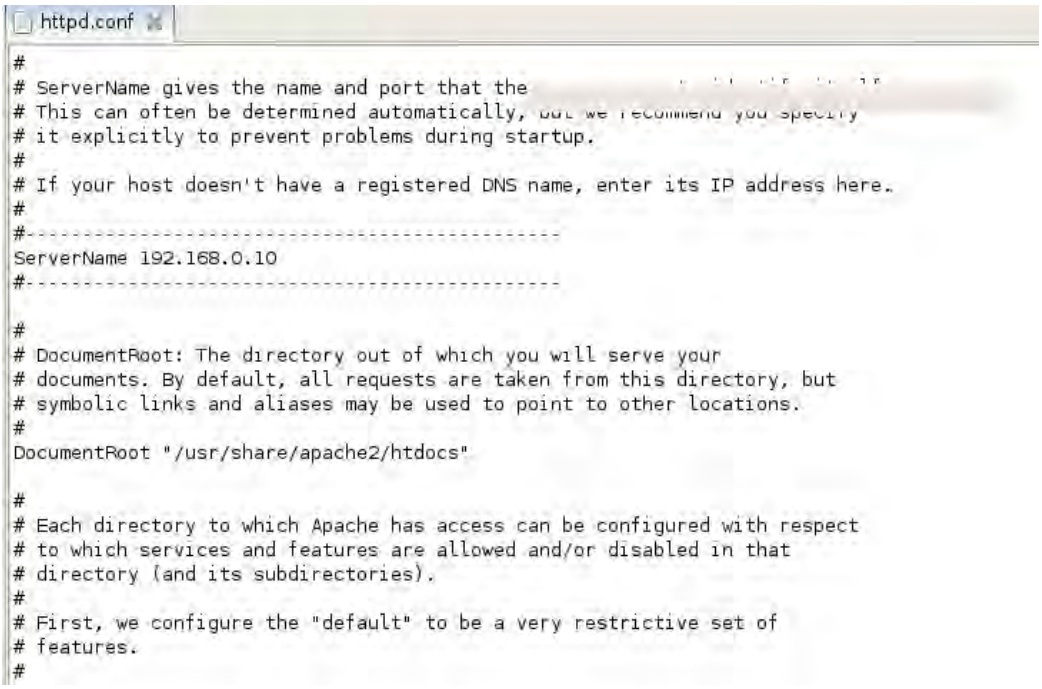
6. Configurar el servidor apache, para esto ir a la carpeta desde la consola cd /etc/apache2

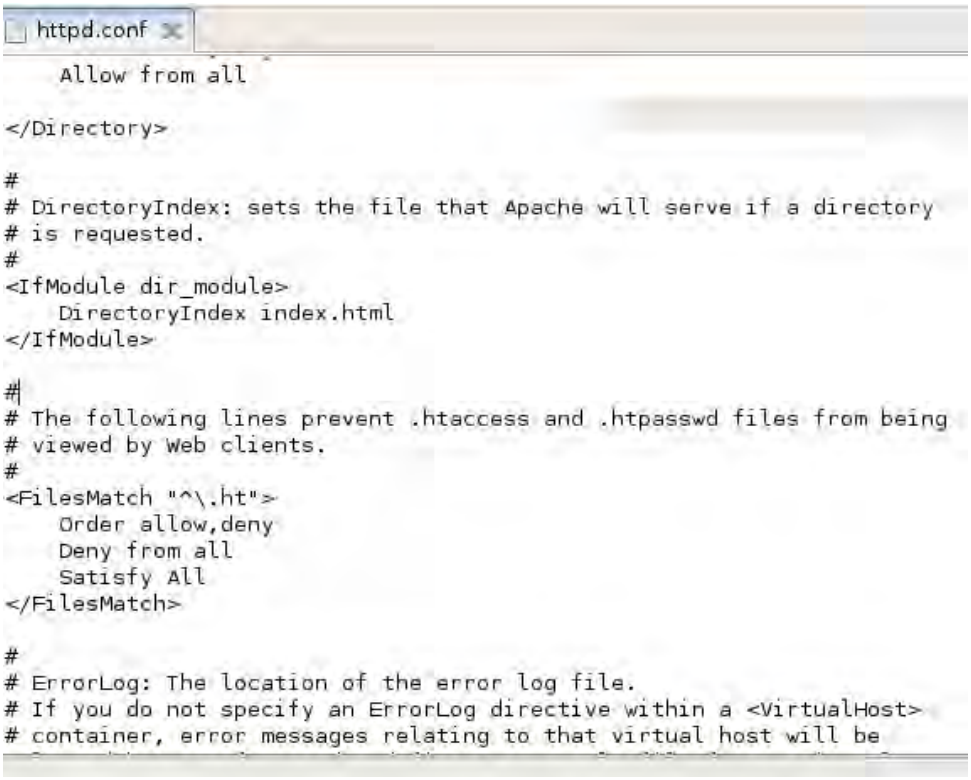
7. Dentro de la carpeta gedit /httpd.conf

8. Dentro del archivo de configuración de apache, se realizara la configuración de la IP.

9. ServerName, es la dirección IP Static que se configuro en el punto 5.

10. DocumentRoot, es el directorio donde se va almacenar la pagina http con el nombre de index.html





11. Ir a la carpeta desde la consola cd /usr/share/apache2/htdocs y guardar la pagina html dentro de la carpeta.

Anexo D. Código HTML

```
<html>
<head>
<title>Detección Arañas</title>
<style type="text/css">
<!--
.Estilo1 {
	color: #FFFFFF;
	font-weight: bold;}
Estilo2 {
	color: #FFFF00;
	font-weight: bold;}
Estilo3 {
	color: #0000FF;
	font-weight: bold;}
Estilo4 {color: #FFFF00}
Estilo5 {color: #FFFFFF}
Estilo6 {color: #FFFF00}
Estilo7 {color: #FFFF00}
-->
</style>
</head>

<body background="Telara.jpg">
<span class="Estilo7">
<script>
var mydate=new Date();
var year=mydate.getYear();
if (year < 1000)
year+=1900;
var day=mydate.getDay();
var month=mydate.getMonth()+1;
if (month<10)
month="0"+month;
var daym=mydate.getDate();
if (daym<10)
daym="0"+daym;

document.write(""+daym+"/"+month+"/"+year+"")
</script>
</span>

<span class="Estilo3"><span class="Estilo4">Hora:</span>
```

```
<script>
</script>
</span>
<span class="Estilo6">
<script>miFecha = new Date()
document.write(miFecha.getHours() + ":" + miFecha.getMinutes() + ":" +
miFecha.getSeconds())
</script> </span>
<center>
<h1 class="Estilo1">IMAGENES DETECCIÓN DE ARAÑAS</h1>
</center>
<p class="Estilo4"><b>LISTA DE IMAGENES</b></p>
<ol>
                <li class="Estilo5">Araña1</li>
                 <li class="Estilo5">Araña2</li>
                 <li class="Estilo5">Araña3</li>
                 <li class="Estilo5">Araña4</li>
                 <li class="Estilo5">Araña5</li>
</ol>
<center>
<h3 class="Estilo2">Imagen # 1</h3>
<IMG SRC="ara.jpg" WIDTH=640 HEIGHT=480 BORDER=3>

<h3 class="Estilo2">Imagen # 2</h3>
<IMG SRC="ara2.jpg" WIDTH=640 HEIGHT=480 BORDER=3>

<h3 class="Estilo2">Imagen # 3</h3>
<IMG SRC="ara3.jpg" WIDTH=640 HEIGHT=480 BORDER=3>

<h3 class="Estilo2">Imagen # 4</h3>
<IMG SRC="ara4.jpg" WIDTH=640 HEIGHT=480 BORDER=3>
</center>
<center>
<h3 class="Estilo2">Imagen # 5</h3>
<IMG SRC="ara5.jpg" WIDTH=640 HEIGHT=480 BORDER=3>
</center>
</body>
</html>
```